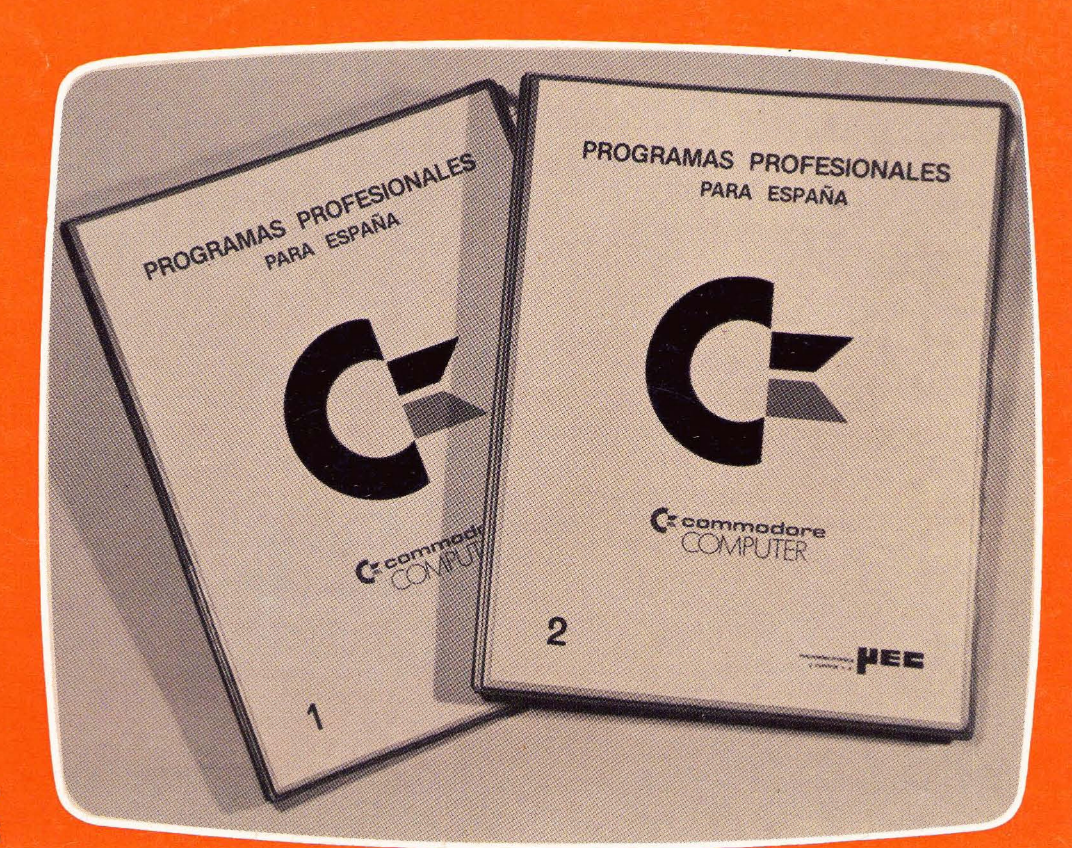
club commodore

Boletín informativo para los usuarios de microordenadores VIC y CBM



Nº/5 febrero 1983

PROGRAMAS VIC-20

Como consecuencia de múltiples llamadas telefónicas recibidas en MICROELECTRÓNICA Y CONTROL preguntando por los programas del VIC damos en este número una lista de ellos junto con algunos datos de interés

Ref.	Programa	P.V.P. Ptas.
	Libro curso de introducción al Basic Parte I Con dos cintas o un disco.	2.500
	Libro curso de introducción al Basic Parte II Con dos cintas o un disco.	2.500
D-1001	**Agenda** Necesita ampliación de 8K.	5.000
D-1002	**QSL** Necesita ampliación de 8K.	3.000
D-1003	**Test Demo**	3.000
D-1004	**Assembler** Necesita ampliación de 3 K.	5.000

CINTAS PARA CASSETTE

Ref.	Programa	P.V.P. Ptas.
C-125	**Hangmath** No necesita ampliación.	1.500
C-128	**Programación lineal** No necesita ampliación.	500
C-129	**Matrices** No necesita ampliación.	500
C-130	**Caja** Necesita ampliación de 16K	1.500
C-131	**Regresiones I** No necesita ampliación.	500
C-132	**Regresiones II** No necesita ampliación.	500
C-133	**Estadística I** No necesita ampliación.	500
C-134	**Estadística II** No necesita ampliación.	500
C-135	**Sistemas** No necesita ampliación.	500
C-136	**Dieta** Necesita ampliación de 8K.	1.500
C-137	**Integración** No necesita ampliación.	500
C-139	**Vicalc** No necesita ampliación.	1.500
C-140	**Skymath** Necesita ampliación de 3K.	1.500
C-141	**Space Division** Necesita ampliación de 3K.	1.500
C-142A	**Interface y programa CW** No necesita ampliación.	25.000
C-142B	**Programa RTTY** No necesita ampliación.	2.500
C-143	**English Language** Necesita ampliación de 8K.	2.000
C-144	**Quiz Master** Necesita ampliación de 8K.	2.000
C-145	**Mastermind** Necesita ampliación de 8K.	3.000
	Además para esta cassette se están preparando los siguientes temas:	
C-145A	**Gastronomía**	1.500
C-145B	**Música**	1.500
C-145C	**Cinematografía**	1.500
C-145D	**Deportes** Etc...	1.500
C-201	**Codemaker** No necesita ampliación.	1.500
C-202	**Wall Street** No necesita ampliación.	1.500
C-203	**Simple Simon** No necesita ampliación.	1.500
C-204	**Damas** No necesita ampliación.	1.500
C-205	**Alien Blitz** No necesita ampliación.	1.500
C-206	**Kosmic Kamikaze** Necesita ampliación de 3 u 8K, dado que hay dos versiones del juego.	1.500
C-207	**Star Wars** No necesita ampliación de memoria.	1.500
C-208	**Amok** No necesita ampliación.	1.500
C-209	**The Alien** Necesita ampliación de 3K.	1.500
C-210	**Invader Fall** No necesita ampliación.	1.500
C-211	**A-MAZ-ING** Necesita ampliación de 3K.	1.500
C-212	**Math-Hurdler, Monster Maze** No necesita ampliación.	1.500
C-213	**Golf** Necesita ampliación de 3K.	1.500
C-215	**VIC GAMES II** No necesita ampliación.	1.500

JUEGOS EN CARTUCHO

Ref.	Programa	P.V.P. Ptas.
1904	**Super Slot** Se juega por teclado o con Joystick.	4.500
1906	**Super Allien** Se juega por teclado o con Joystick.	4.500
1907	**Júpiter Lander** Se juega por teclado.	4.500
1908	**Draw Poker** Se juega por teclado.	4.500
1901	**Avenger** Se juega por teclado o con Joystick.	4.500
1909	**Road race** Se juega por teclado.	4.500

PROGRAMAS PROFESIONALES

Ref.	Programa	P.V.P. Ptas.
Z0001	**Contabilidad S/V-20** **Stock-VIC**	
V0501	**Control películas para Video-Clubs**	
Z0601	**VIC/Entrapunt**	

Nota: Para más detalles sobre estos programas consultar.

NOVEDADES ENERO 1983

PROGRAMAS EN CINTA:

Ref.	Programa	P.V.P. Ptas.
C-146	**Matemáticas 1** (nivel BUP) Necesita ampliación de 8 ó 16K.	2.000
C-216	**Escape** No necesita ampliación.	1.500
C-217	**Biocomp** Necesita ampliación de 3K.	2.000
C-218	**Cubo VIC** No necesita ampliación.	1.500
C-219	**Type a tune** No necesita ampliación.	2.000

PROGRAMAS EN CARTUCHO:

Ref.	Programa	P.V.P. Ptas.
1911	**The sky is failling** Es necesario utilizar Paddle.	4.500
1902	**Star battle** Se juega con teclado o con Joystick.	4.500
1919	**Sargon II chess** Se juega con teclado o con Joystick.	4.500
1924	**Omega race** Se juega con Joystick.	4.500
C-400	**VIC FORTH**	consultar
C-401	**VIC STAT**	consultar
C-402	**VIC GRAPH**	consultar
1912	**Mole attack** Sólo para teclado.	4.500
1909	**Radar ratrace** Se juega con teclado o con Joystick.	4.500
C-403	**WORDCRAFT-VIC** (compatible series CBM 8000 y CBM 4000)	42.500
C-404	**VIC SCREEN MASTER**	10.000

PROGRAMAS EN DISCO

Ref.	Programa	P.V.P. Ptas.
D-1005	**English Language** Necesita ampliación de 8K.	2.500
D-1006	**Quiz master** Necesita ampliación de 8K.	2.500
D-1007	**Matemáticas I** (nivel de BUP) Necesita ampliación de 8K.	2.500
D-2000	**VIC WRITER** Necesita ampliación de 8K.	13.000
D-2001	**SIMPLICALC** Necesita ampliación de 8K.	13.000
D-2002	**VIC FILE** Necesita ampliación de 8K.	13.000

Depósito Legal Sep. B. 2133-1958 - Altés, s.l., Caballero 87, Barcelona

EDITORIAL

la tercera solución commodore

Hemos editado, recientemente, las Carpetas de Programas Profesionales para equipos COMMODORE, dos de las cuales ilustran la portada de este número. Aunque orientadas a servir como herramientas de trabajo a nuestros distribuidores, están pensadas para facilitar la consulta y elección de las aplicaciones necesarias para cada usuario en particular.

Después de más de 4.000 instalaciones en España, nuestra experiencia nos dice que la mayoría de usuarios lo que realmente buscan no es un "ordenador" sino una solución, y ésta consta de tres partes insustituibles: el equipo, los programas, y la asistencia.

En cuanto al equipo, poco vamos a extendernos ya que es suficientemente conocido por su facilidad de manejo, su solidez y la baja tasa de fallos producidos. Sólo resaltar la continuidad y progresión mantenidas por COMMODORE, desde su famoso PET, lo que le coloca en el número uno en ventas, no sólo en nuestro país sino también en el Mundo. En este momento posee la más extensa gama de equipos con los que podemos configurar sistemas con capacidades desde 32 K hasta 96 Kbytes en RAM y desde 170 K hasta 9 Mbytes en disco.

Los otros dos puntos, programas, y asistencia, van muy relacionados con la figura del distribuidor COMMODORE, persona que contacta, atiende, repara, programa, instala, etc.; a través de sus distintos departamentos. Gracias a que TODOS nuestros distribuidores disponen de departamentos técnicos de programación y mantenimiento, podemos dar el mejor y más repartido servicio.

En el entorno de la informática las distintas marcas plantean dos líneas bien distintas: las que opinan que el cliente debe adaptarse a los programas estándar y las que creen que para cada cliente es necesario realizar la aplicación a medida (llaves en mano).

Las primeras acostumbran a tener 4 o 5 programas, solamente, cuya cualidad de ser estándar quizá no sea el momento de evaluar. Mientras que las segundas invierten en cada instalación 3 ó 4 meses, en el mejor de los casos, en la realización de los paquetes de programas.

Debido a la preparación y organización del equipo técnico de nuestra Red de Distribuidores, COMMODORE puede afrontar cualquiera de ellas y, con la presentación de las Carpetas de Programas, además quiere aportar una tercera solución, suma de las ventajas de las anteriores y con los menos defectos posibles: eliminar la necesidad de que el cliente se adapte al programa y reducir el tiempo de espera para programas "particulares" de nuestra biblioteca, a una pura puesta en marcha.

A este fin se han creado las Carpetas de Programas para dar a conocer qué programas existen, que ya funcionen y qué procesos detallados efectúan, siendo una recopilación de la extensa biblioteca de programas, tanto realizados por Microelectrónica y Control S.A., como por nuestros distintos distribuidores, en y para España.

Están a disposición de cualquier usuario de sistemas de gestión que quiera ampliar el rendimiento de su equipo o para aquellos que su primer contacto con COMMODORE ha sido el VIC-20 y necesitan "algo" de más envergadura para su pequeño negocio.

(Departamento de Difusión
Microelectrónica y Control S.A.)

PROGRAMAS PROFESIONALES (Tomo I)

Ref.	Título
DMA-401	Base de datos
DMA-402	Contabilidad
DMC-413	Wordcraft
DMC-461	VisiCalc
DMA-467	Assembler
DMA-801	Base de Datos
DMA-802	Contabilidad
DMA-803	Gestión Comercial
DMA-804	Paso de Gestión Comercial a Contabilidad
DMA-807	Etiquetas
DDA-810	Nóminas
DMC-813	Wordcraft
DMC-815	Contabilidad 4N
DDS-817	Renta-8X
DDS-818	Fincas.H
DDC-819	SADIC
DMC-861	VisiCalc
DMA-867	Assembler
DMC-868	Master
EMC-871	Edex
DMC-877	Protector
DIC-8302	Dirección
DIC-8303	Facturación de Industrias Cárnicas
DIC-8306	Control de Socios
DIC-8309	Producción Industrial

PROGRAMAS PROFESIONALES (Tomo II)

Ref.	Título
DDC-403	Gestión Comercial
DDA-409	Colegios
DMC-416	Gabinete Odontológico
DDC-476	Petspeed
DDA-809	Colegios
HMC-872	Placa de Alta Resolución
DDC-876	Petspeed
DIA-4301	Compra-venta de vehículos
DIA-4302	Facturación con báscula
DIA-8301	Consignatarios de pescado
DIC-8304	Componentes
DIA-8305	Stock zapatería
DIA-8307	Boutiques
DIS-8308	Hostelería
DIC-8310	Agentes de quinielas
DIC-8311	Agenda
DIA-8312	Medios de Publicidad
DIC-8313	Macro-Stocks
DIA-8314	Control de Producción y Personal
DIC-8315	Empresas de Autoventas
DIC-8316	Facturación Empresas de Limpieza
DIC-8317	Rentas
DIC-8318	Administración Fincas
DIA-8319	Facturación con báscula
DIC-8320	Colegios Profesionales

VENTANA CBM

viaje a las profundidades de la zona de programas

por **JOAN CARLES SAMARANCH**

El tema de este mes trata de cómo trabaja el sistema con los programas. Cuando escribimos una instrucción cómo la «interpreta» el BASIC. Para entender todo lo que vamos a comentar nos referiremos a la tabla 1.

ABREVIACIONES

Lo de las abreviaciones es poco conocido pero muy práctico para poder incrementar la velocidad de programación y para poder construir líneas de programa de más de 80 columnas (referente al listado).

Por ejemplo, la instrucción PRINT se puede abreviar escribiendo «?». Escribamos la línea: 10 ? y listemos el programa. Veremos 10 PRINT.

Para el resto de instrucciones indicamos, en la tabla 1, en minúscula el carácter a escribir en mayúscula (+ SHIFT) que, en el caso de trabajar en caja alta (mayúsculas/gráficos), aparecerá el gráfico correspondiente en pantalla. En caja baja (minúsculas/

TABLA 1

Instrucción	Abreviación	Token Decimal	Token Hexadecimal	Instrucción	Abreviación	Token Decimal	Token Hexadecimal
END	En	128	80	SPC(	Sp	166	A6
FOR	Fo	129	81	THEN	Th	167	A7
NEXT	Ne	130	82	NOT	No	168	A8
DATA	Da	131	83	STEP	STe	169	A9
INPUT #	In	132	84	+		170	AA
INPUT	INp	133	85	—		171	AB
DIM	Di	134	86	*		172	AC
READ	Re	135	87	/		173	AD
LET	Le	136	88	↑		174	AE
GOTO	Go	137	89	AND	An	175	AF
RUN	Ru	138	8A	OR	Or	176	B0
IF	If	139	8B	>		177	B1
RESTORE	REs	140	8C	=		178	B2
GOSUB	GOs	141	8D	<		179	B3
RETURN	REt	142	8E	SGN	Sg	180	B4
REM	REm	143	8F	INT	INt	181	B5
STOP	St	144	90	ABS	Ab	182	B6
ON	On	145	91	USR	Us	183	B7
WAIT	Wa	146	92	FRE	Fr	184	B8
LOAD	Lo	147	93	POS	POs	185	B9
SAVE	Sa	148	94	SQR	Sq	186	BA
VERIFY	Ve	149	95	RND	Rn	187	BB
DEF	De	150	96	LOG	LOg	188	BC
POKE	Po	151	97	EXP	Ex	189	BD
PRINT #	Pr	152	98	COS	COs	190	BE
PRINT	?	153	99	SIN	Si	191	BF
CONT	Co	154	9A	TAN	TAn	192	C0
LIST	Li	155	9B	ATN	At	193	C1
CLR	Cl	156	9C	PEEK	Pe	194	C2
CMD	Cm	157	9D	LEN	Le	195	C3
SYS	Sy	158	9E	STR$	STr	196	C4
OPEN	Op	159	9F	VAL	Va	197	C5
CLOSE	CLo	160	A0	ASC	As	198	C6
GET	Ge	161	A1	CHR$	Ch	199	C7
NEW	NEw	162	A2	LEFT$	LEf	200	C8
TAB(	Ta	163	A3	RIGHT$	Ri	201	C9
TO	To	164	A4	MID$	Mi	202	CA
FN	Fn	165	A5				

mayúsculas), el texto a escribir es precisamente al revés de lo indicado en la tabla 1, cambiando las mayúsculas por minúsculas y viceversa.

«TOKEN'S»

En el ejemplo anterior el sistema no guarda las letras: P, R, I, N y T, correspondientes a la instrucción si no que guarda un solo byte, llamado «token», que codifica el tipo de instrucción. En la tabla 1 vemos el número que corresponde a cada instrucción. Debemos resaltar que existen a partir del valor 128 ($80), o sea el bit 2^7 a uno.

ESTRUCTURA INTERNA

El «programa» de ejemplo es:

```
10 REM ***** ****
```

Después de crear esta línea vamos al monitor: SYS1024, y visualizamos el principio de la zona de programas con la instrucción:

```
.M 0400 0410
```

apareciendo lo siguiente:

```
.: 0400 00 11 04 0A 00 8F 2A 2A
.: 0408 2A 2A 2A 2A 2A 2A 2A 2A
.: 0410 00 00 00 AA AA AA AA AA
```

Vamos a analizar estos valores. En la posición 1024 ($0400 en hexadecimal) tenemos el cero inicial; en 1025 y 1026 tenemos la dirección donde empieza la siguiente linea de programa; en el formato peso bajo/peso alto, en este caso $0411; en 1027 y 1028, con el mismo formato, el número de línea de programa que le corresponde, $00A0 igual a 10 en decimal; en 1029 está el «token» correspondiente a la instrucción REM que es $8F; de 1030 a 1039 los diez asteriscos, $2A y, por último, un cero indicando el final de línea. Por ser un programa de una sola línea, los dos bytes siguientes, que corresponderían a la dirección de inicio de la siguiente línea, son cero, indicando fin de programa (el siguiente byte ya correspondería a la zona de variables).

GENERALIZACIÓN

Lo que hemos visto con un ejemplo sencillo puede extrapolarse a cualquien programa. En la figura 1 vemos un gráfico, general, de la estructura de un programa.

El significado de las abreviaciones allí expuestas son: LINK AD, dirección donde empieza la siguiente línea de programa; LINE NO, número de línea BASIC asignada, y TEXT, instrucciones asociadas a esta línea.

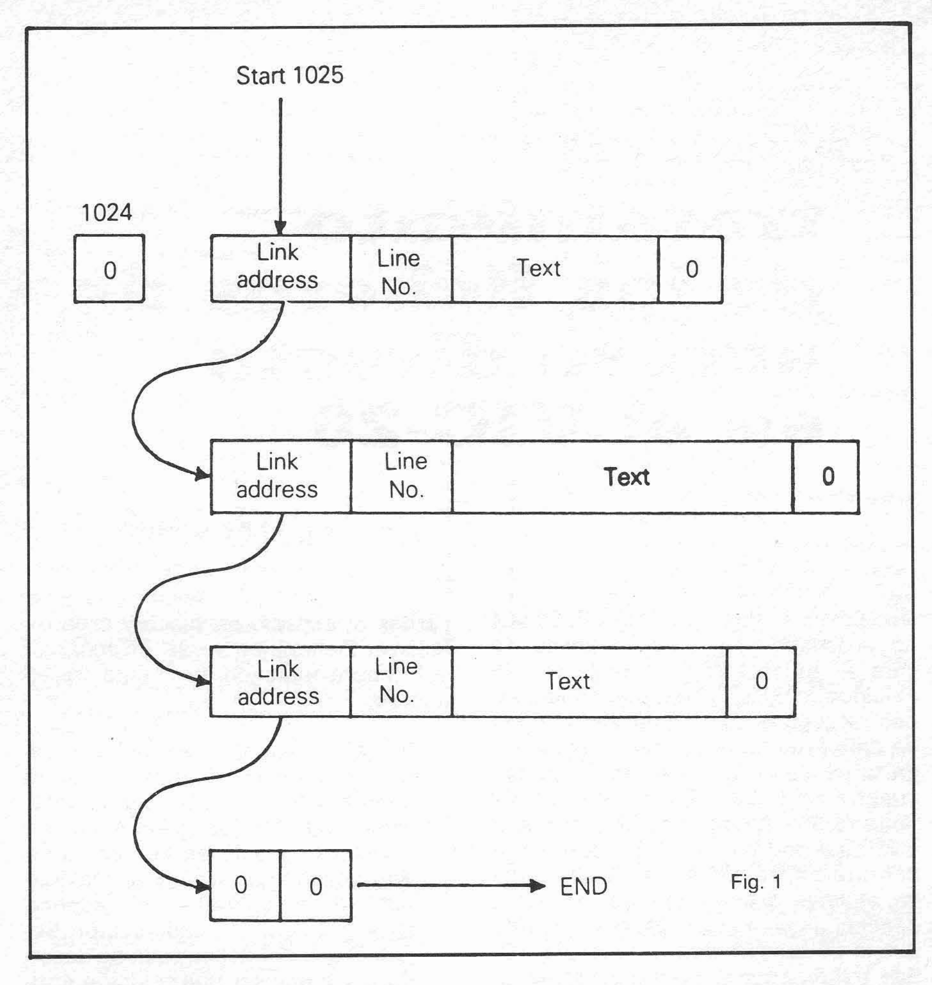


Fig. 1

En la última línea, lo que correspondería al LINK AD está a cero para indicar el fin de programa, fácilmente reconocible por los tres ceros que aparecen. ■

TRANSFERENCIA DE PROGRAMAS ENTRE EL VIC Y LOS ORDENADORES CBM 4000 Y 8000

Existen dos métodos de transferencia de programas entre el VIC-20 y los ordenadores CBM 4000 y 8000. Debe entenderse que la compatibilidad (no se trata de esa compatibilidad de la que tanto se habla últimamente en Madrid, sino de la posibilidad de que una máquina pueda leer programas y datos grabados por otra) es total a nivel de cassette entre todos los equipos y a través de disco solamente entre el VIC y la serie 4000. Aparte debe vigilarse la ejecución de un programa en una máquina y escrito en otra pues es muy posible que los POKEs no funcionen de la misma forma. De todas maneras con los métodos que damos a continuación se pueden listar programas de VIC en otros equipos. El problema principal consiste en que, a diferentes configuraciones de memoria, el comienzo de Basic (START OF BASIC) cambia de forma automática en el VIC mientras que no se mueve en los equipos 4000 y 8000. En el caso de grabar el programa con el cartucho de 3K en el VIC los programas entran sin ningún problema en los equipos CBM. Sin embargo, en el caso de haberlo cargado desde un VIC sin expansión podemos hacer lo siguiente: 1) CARGAR EL PROGRAMA; 2) ESCRIBIR 10REM(RETURN) — SIN NINGÚN ESPACIO —; 3) POKE 1025,7: POKE 1026,16 (RETURN); 4) ELIMINAR LA LINEA 10 (PULSAR 10 Y RETURN); 5) POKE 43,(PEEK(43)-12):CLR (RETURN).

Todo ello debe hacerse resistiendo la tentación de LISTar hasta haber terminado todo el proceso. En caso de haber cargado el programa desde un VIC con más de 8K, los pasos 3 y 5 deben ser: 1) LO MISMO; 2) LO MISMO; 3) POKE 1025,7: POKE 1026,18; 4) LO MISMO; 5) POKE 43,(PEEK(43)-14):CLR.

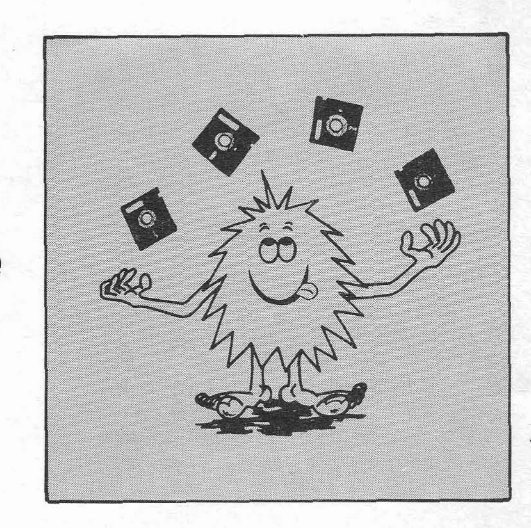
Este método tiene el inconveniente de que necesitamos recordar en qué configuración de VIC se ha grabado el programa. En el método que damos a continuación se resuelve este problema. Antes de cargar el programa del VIC en el CBM pulsar: NEW (RETURN). FOR J=0 TO 2:POKE 4096+J,0:POKE4608+J,0:NEXT J (RETURN).

Cargar el programa y pulsar: LIST (RETURN).

Si el programa aparece ya hemos terminado el proceso; si no es así pulsar: POKE41,16:LIST (RETURN).

Si el programa sigue sin aparecer pulsar: POKE 41,18:LIST (RETURN).

tratamiento de los ficheros de acceso directo en el VIC-20



por **PERE CARBÓ**

A pesar de que tal vez el VIC-20 sea un ordenador personal enfocado de cara a aplicaciones recreativas, de cálculos técnicos o uso doméstico, etcétera, y en la mayoría de estos tipos de aplicaciones no vamos a necesitar de la ayuda de ficheros que soporten nuestra información, y menos aún de ficheros de acceso directo, que son a los que me voy a referir, me voy a permitir en las siguientes líneas revelar algunas experiencias al respecto, deseando que a algún usuario que disponga de la unidad de disquette y desee trabajar con este tipo de ficheros, le puedan ser útiles.

Como referencia, el capítulo 7 del MANUAL DE LA UNIDAD DE DISCO en sus páginas 53 a la 59, trata de explicar el uso de estas técnicas. De allí están tomadas las subrutinas que más tarde vamos a ver y ha servido de base, para elaborar este trabajo, que no pretende ser más que una simple documentación anexa que amplíe, en cierto modo, lo allí expuesto, a fin de optimizar al máximo esta «herramienta» que nos ofrece el sistema operativo de disco del VIC-20.

PREMISAS - 1

El uso de estos ficheros que «a priori» puede parecer algo engorroso, es, sin embargo, totalmente eficaz si se les da el tratamiento adecuado. Codificando las instrucciones oportunas que a continuación veremos y ayudándonos de una pequeña subrutina de cálculo, obtendremos una dirección física de disco (Pista y Sector) donde registraremos la información. Más tarde, para acceder a ella, tan sólo tendremos que recurrir a la misma subrutina de cálculo.

En primer lugar, nos interesa reservarnos el espacio en disco y crear el fichero. Deberá preverse el volumen del fichero teniendo en cuenta varios puntos:

1.° Por el tratamiento del fichero, cada vez que «escribamos» en él, lo vamos a hacer en un sector diferente, por lo que ganaremos en cuanto a espacio en la medida en que ocupemos el número máximo posible de los 250 bytes disponibles que contiene cada sector. Según el uso del fichero que se desee hacer, puede ser una solución óptima recopilar varios registros lógicos en un solo registro físico.

2.° Aunque se trate de un fichero en el que no vayamos a hacer la carga inicial de los datos de una sola vez sino que vamos a hacerlo paulatinamente, debemos crearlo igualmente en función del volumen total previsto para así reservarnos los sectores correspondientes donde más tarde irá la información. De no ser así, al acceder a una dirección del disco fuera del espacio que previamente hemos reservado, podríamos «machacar» lo que hubiera en esa dirección (programas, otros ficheros), lo cual nos daría más de un problema.

PREMISAS - 2

Por otra parte, vamos a necesitar paralelamente al fichero directo, un segundo fichero que nos sirva de índice del primero. Este fichero índice será de organización secuencial y podrá ser diferente en función de como vayamos a tratar el de acceso directo.

Si a pesar de tratarse de un fichero de acceso directo, la carga inicial de éste la vamos a hacer de un modo secuencial, bastará con que el fichero índice contenga un solo registro con el número correspondiente al último registro grabado en el fichero directo. De este modo, cuando queramos leer de él, únicamente deberemos comprobar que no se intenta acceder a un registro mayor que el último grabado y, del mismo modo, cuando queramos grabar más datos, nos servirá de puntero para saber cuál debe ser el siguiente número de registro.

Un pequeño ejemplo de esta utilización, podría ser un fichero maestro de socios, en el cual los vamos a ir registrando secuencialmente según se produzca el alta. Vamos a hacer coincidir el número de asociado con el número del registro que ocupan en el fichero de forma que el socio n.° 1 ocupará el primer registro, el 2.° socio el segundo registro y así sucesivamente. La creación de este fichero sería secuencial aunque anteriormente hayamos creado el fichero en función de las previsiones, para reservarnos el espacio. Así, pues, en el fichero índice tendremos el número del último asociado. Cuando queramos dar una nueva alta, el socio tendrá el número que nos dé el fichero índice + 1. Sin embargo, el tratamiento posterior del fichero, va a ser directo. A la hora de hacer una consulta o una modificación nos vamos a referir en concreto al socio (o registro) 28, 113, 4, etc., y la única precaución que deberemos tener es comprobar que el número de asociado al que vamos a acceder no sea mayor que el índice.

PREMISAS - 3

Sin embargo, si la carga inicial también va a ser aleatoria, entonces este

procedimiento no es válido. Una manera de establecer un índice, sería creando un fichero secuencial con tantos registros de un solo byte como registros vaya a tener el fichero directo. Podemos inicializar el fichero índice con todos los registros a «0» y cada vez que grabemos un registro en el fichero directo mover un «1» a su registro correspondiente en el fichero índice. A la hora de acceder a un registro, bastará acceder primero al índice y conocer su estado para saber si verdaderamente existe o no.

Para optimizar, en cuanto a tiempo, es aconsejable haber cargado previamente los registros del fichero índice en una tabla de memoria sobre la cual podemos trabajar durante la ejecución del programa y, al final, grabarla en disco si es que ha sufrido modificaciones. Con ello ganaremos tiempo y ahorraremos accesos a discos.

En las páginas 55/59 del «Manual Floppy» hay un programa de ejemplos en el que se emplea este procedimiento.

TRATAMIENTO - 1

Para trabajar con fichero de acceso directo, necesitamos hacerlo por medio de un área intermedia a la cual le damos el tratamiento de fichero. Es decir si pretendemos grabar información en un fichero, primero la vamos a llevar a esa área intermedia y, desde allí, la vamos a grabar definitivamente en disco. Y exactamente igual para leerla: del disco la movemos al área intermedia y de allí la recogemos para tratarla.

El esquema de la fig. 1 pretende aclarar un poco el contenido de las líneas anteriores.

Pues bien, sentados ya frente a nuestro «Micro», el primer paso será abrir el fichero de trabajo y el área intermedia que, como hemos dicho, le damos tratamiento de fichero. Deberemos abrir también el fichero índice

```
10 OPEN 5, 8, 15
20 OPEN 4, 8, 3, "#"
30 OPEN 2, 8, 2, "0:ÍNDICE, S, R"
```

TRATAMIENTO - 2

Donde:

5-4-2 = NÚMERO LÓGICO DE FICHERO
8 = TIPO DE PERIFÉRICO (8 = FLOPPY, 1 = CASSETTE)
15-3-2 = NÚMERO DE CANAL
0 = PRIMER DRIVE FLOPPY (Si existiera un 2.° sería 1)
ÍNDICE = NOMBRE DE FICHERO
S = TRATAMIENTO SECUENCIAL
R = READ (Tratamiento de lectura, W = WRITE)

Este paso sería común para cualquier tratamiento que le fuéramos a dar al fichero pero, a partir de ahora, va a variar en función de que queramos acceder a él para leer o simplemente accedamos para grabar datos. En cuanto al fichero índice, hemos visto cómo se abre, pero no vamos a seguir con su tratamiento. Éste sería totalmente secuencial y dependiendo, como hemos dicho, del fichero directo.

Acceso de lectura

En primer lugar, vamos a tener que utilizar la subrutina de cálculo que nos dirá la pista y el sector donde está el registro que estamos buscando (ver subrutina de dirección física), y con los datos que nos devuelva, nos vamos a la información y la moveremos al área intermedia para, desde allí, recogerla y posteriormente tratarla.

```
40 PRINT # 5, "B-R:" 3;0;NP;NS
```

Donde:

5 = Número lógico del fichero de datos
"B-R:" = Comando de sistema operativo que significa Buffer Read (lectura intermedia)

TRATAMIENTO - 3

3 = Número de canal del área intermedia
0 = Número del «Floppy» (disco flexible)
NP = Variable recogida de la subrutina del cálculo de dirección física que especifica el NÚMERO DE PISTA
NS = Variable recogida de la subrutina de cálculo de dirección física que especifica el NÚMERO DE SECTOR

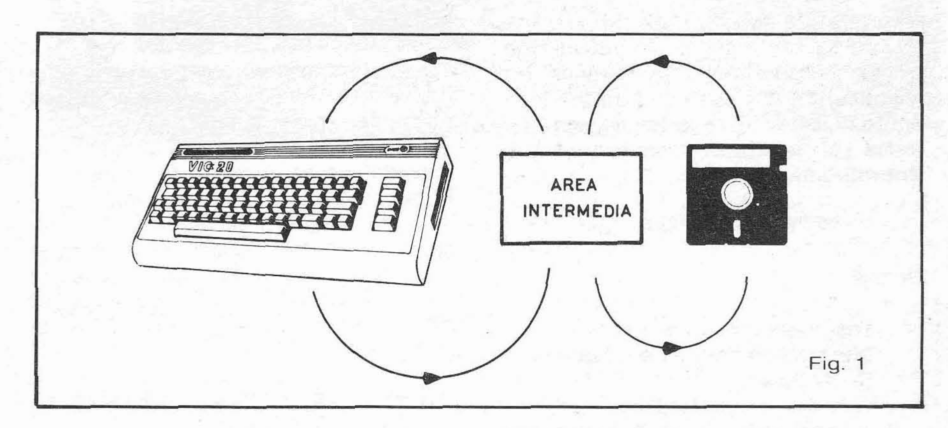


Fig. 1

El siguiente paso es mover el puntero intermedio que controla la transmisión de datos.

```
50 PRINT # 5, "B-P:" 3;1
```

Donde:

5 = Número lógico del fichero de datos
"B-P:" = Comando de sistema operativo que significa «Buffer Pointer» (puntero intermedio)
3 = Número de canal del área intermedia
1 = Posición del puntero

Y, por último, recogemos la información del área intermedia, de la siguiente forma:

```
50 INPUT # 4,A$,B,C
```

Donde:

4 = Número lógico del área intermedia
A$,B,C = Variables

(pasa a la pág. siguiente)

tratamiento de los ficheros de acceso directo en el VIC-20

(viene de la pág. anterior)

TRATAMIENTO - 4

Acceso de escritura

En el tratamiento de escritura, varía el orden de las instrucciones. En primer lugar deberemos mover el puntero intermedio. Para dar una idea de lo que se pretende con esta instrucción, me atrevería a compararla con el dueño de una casa que da una fiesta, cuando presenta a dos de sus invitados (el fichero de trabajo y el área intermedia).

```
40 PRINT # 5 "B-P:",3;1
```

Donde:

Los parámetros son los mismos que cuando hacemos el acceso de lectura

A continuación, vamos a llevar al área intermedia la información que queremos grabar.

```
50 PRINT # 4,A$;B;C;CHR$(13)
```

Donde:

4 = Número lógico del área intermedia
A$,B,C = Variables
CHR$(13) = Marca del «Carriage return» (Fin de registro)

Y, por último, registraremos la información definitivamente en el fichero, pero antes deberemos pasar por la subrutina de cálculo de dirección para que en función del número de registro, le asigne un lugar en el fichero a nuestra información.

TRATAMIENTO - 5

```
60 PRINT # 5, "B-W:" 3;0;NP;NS
```

Donde:

Los parámetros son los mismos que en la línea 40 del acceso de lectura, con la salvedad de que el comando "B-R:" («Buffer Read»), es sustituido por "B-W:" («Buffer Write») (escritura intermedia)

SUBRUTINA DE CÁLCULO DE DIRECCIÓN FÍSICA

A continuación, vamos a ver la subrutina que nos calcula la dirección física del registro en el disco.

El único parámetro que le damos de entrada, es el número de registro, y la subrutina nos devuelve dos variables (NP, número de pista) y (NS, número de sector), situación física que le corresponde ocupar en el disco al determinado registro. La variable NR es la que contiene el número de registro.

```
100 IF NR 358 THEN F1=0:F2=22:
    F3=1:GOTO 140
120 IF NR 357 AND NR 471 THEN
    F1=357: F2=20: F3=19: GOTO
    140
120 IF NR 471 AND NR 580 THEN
    F1=471: F2=19: F3=25: GOTO
    140
130 IF NR 580 THEN F1=580: F2=18:
    F3=31
140 NP= INT (((NR-F1)-1)/(F2-1)) F3
150 NS= NR-F1-(NP-F3) F2 (NP-F3-1)
```

Los registros deberán ir siempre numerados desde 1 hasta 664 (capacidad máxima del disquette); y el siguiente cuadro, extraído del manual del floppy, muestra la distribución general de éstos en el mismo:

Número Pista	Sector	Sectores/Pista	Núm. Registro
1 a la 17	0 al 20	21	1 al 357
18 a la 24	0 al 80	19	358 al 471
25 a la 30	0 al 17	18	478 al 579
31 a la 35	0 al 16	17	586 al 664

TRATAMIENTO DE ERRORES

El sistema operativo de disco del VIC-20 nos permite el tratamiento de errores en los accesos a disco. Mediante la subrutina que veremos a continuación podemos tratar de controlar los errores que se puedan dar durante la ejecución de un programa, en cuanto a lo que accesos a ficheros se refiere, con lo que de alguna manera, aunque el error ya es inevitable, podemos evitar esas detenciones bruscas del programa que nos «desmontan» la pantalla y nos aparece esa famosa palabra ya familiar para todo aquel que ha programado más de dos días de «ERROR».

```
200 INPUT # 15, NE,ME$,PE,SE
210 IF NE = 0 THEN RETURN
220 PRINT «ERROR DE DISCO »;
    NE;ME$;PE;SE
230 INPUT"CONTINUO "; Y$:IF Y$=
    «SI» THEN RETURN
240 STOP
```

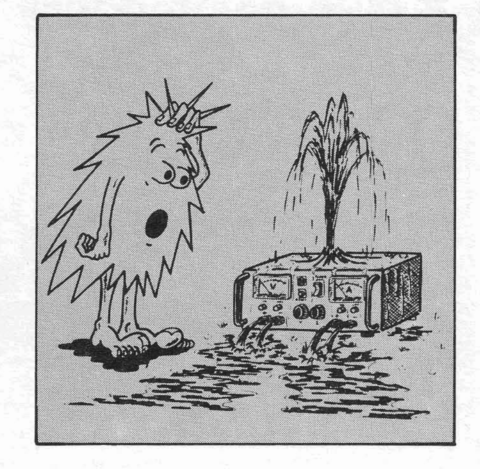
Donde:

15 = Número de canal
EN = Número de error (ver manual del floppy)
ME$ = Mensaje o descripción del error
PE = Pista donde se ha producido el error
SE = Sector donde se ha producido el error

Con esta subrutina, recogemos información del sistema y comprobamos el número de error (línea 210). Si éste es igual a 0, es que no ha habido error y vuelve el programa a su ejecución normal. Si lo hubiese, nos daría por pantalla los datos referentes al error (número, descripción, pista y sector) y nos daría la opción de seguir o detener la ejecución del programa. ■

ELECTRÓNICA

diseño de fuentes de alimentación



por **R. PARDO**

Este programa calcula una fuente de alimentación estabilizada electrónicamente con zener y un transistor regulador serie. Con él se obtienen todos los parámetros que se necesitan (tensión en el secundario del transformador, fusible del primario, características de los diodos del puente rectificador, características del condensador de filtro, zener y sus componentes auxiliares, y el circuito del amplificador de error y del transistor serie).

Los datos que pide el programa son:

UAC (V): tensión del primario del transformador (tensión de red).

UDC (V): tensión de salida del estabilizador.

I. SAL (ma): intensidad de salida del estabilizador.

FUSIBLE (L/R): elige fusible lento o rápido para el primario del transformador.

F. RED (Hz): frecuencia de red.

U. zener (V): tensión del diodo zener.

W. zener (mW): potencia máxima de disipación del zener (dada por el fabricante).

BETA TR. SERIE: B (HFE) del transistor serie.

Todos los resultados vienen dados en:

— miliamperios si se trata de intensidades.
— voltios si se trata de tensiones.
— Vatios si se trata de potencias de disipación.
— Microfaradios si se trata de capacidades.
— Kilo-ohmios si se trata de resistencias.

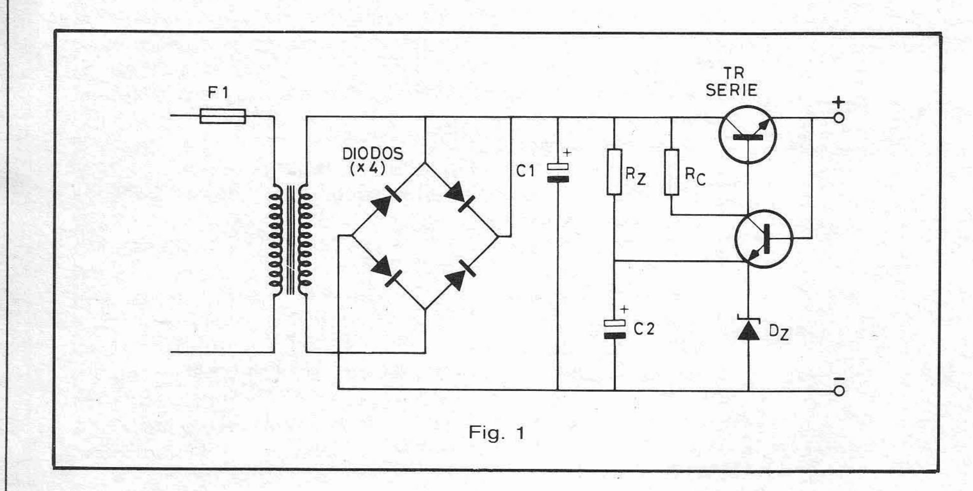


Fig. 1

En todos los casos es aconsejable que, si se intenta llevar a la práctica el circuito, se dote al transistor serie de un elemento refrigerador (un disipador de aletas, por ejemplo).

Las líneas 10-90 presentan el programa y en las líneas 100-210 se introducen los datos con los que se va a trabajar.

En las líneas 220-300 se calculan y

(pasa a la pág. siguiente)

(viene de pág. anterior)

```
ready.

10 rem alimentacion
20 rem r. pardo
30 rem club commodore
40 rem 16.10.1982
50 rem
60 rem
70 rem
80 rem
90 poke36869,peek(36869)+2
100 print"■■FUENTE DE ALIMENTACION■
110 print:input"Uac(V)";u1
120 print:input"Udc(V)";u2
130 print:input"Isal(mA)";i1
140 print:print"Fusible(L/R)"
150 geta$:ifa$="l"thenl=5:goto180
160 ifa$="r"thenl=10:goto180
170 goto150
180 print:input"F. red(Hz)";f
190 print:input"U. zener(V)";uz
200 print:input"W. zener(mW)";wz
210 print:input"Beta tr. serie";bt
220 rem
230 ue=u2/.9:fs=((l*(u1/ue))/i1)*1000
240 print"■FUSIBLE PRIMARIO"
250 print:print"If="fs"mA"
260 gosub640
270 rem
280 print"■TENSION SECUNDARIO
290 print:print"Us="ue"Volts"
300 gosub640
310 rem
320 rm=u2/i1:im=i1/2
330 ur=ue*sqr(2)
340 print"■CARACTERISTICAS DIODOS"
350 print:print"If="im"mA"
360 print:print"Ur="ur"Volts"
370 gosub640
380 l=2*f
390 t=1/l:cr=(3*t)/rm:c=cr*2000
400 uc=ur+((ur*40)/100)
410 print"■CALCULO CONDENSADOR"
420 print:print"C1="c"uF"
430 print:print"Uc1="uc"Volts"
440 gosub640
450 rem
460 iz=wz/uz:rl=(ur-u2)/iz
470 wr=(ur-u2)*iz:ib=i1/bt:i4=2*ib
480 w1=i4*u2:ifw1>=wzthenprint"■■EL ZENE
R DISIPA DEMA- ■■    ■SIADA POTENCIA■":g
oto670
490 rn=(ur-u2)/ib:cl=(3*t)/rl:cs=cl*2000
500 print"■CALCULO ESTABILIZADOR"
510 print:print"Rz="rl"kOhms
520 print:print"Wrz="wr"mW"
530 print:print"C2="cs"uF"
540 print:print"Uc2="uc"Volts
550 print:print"Rc="rn"kOhms"
560 gosub640
570 print"■1-Repeticion datos
580 print:print"2-Nuevo proyecto
590 print:print"3-Fin de proyecto
600 print:print"Pulse 1,2 o 3
610 getb$:ifb$=""then610
620 k=val(b$):onkgoto240,100,630
630 poke36869,peek(36869)-2:print"■":end
640 print:print"Pulse una tecla
650 gets$:ifs$=""then650
660 return
670 print:print"Tiene dos opciones"
680 print:print"1-Tr. serie de beta    ■m
as elevada"
690 print:print"2-Zener mas potente
700 print:print"Pulse 1 o 2
710 getl$:ifl$="1"then740
720 ifl$="2"then750
730 goto710
740 print"■":input"Beta";bt:goto460
750 print:"■":input"W. zener(mW)";wz:got
o460

ready.
```

presentan los datos del fusible secundario del transformador. Las características de los diodos se calculan en las líneas 310-370, el condensador de filtro en las líneas 380-440 y el resto del estabilizador en las líneas 450-560. Las líneas 570-630 y las 640-660 son unas rutinas para finalizar el programa en el primer caso y para seguir con la pantalla siguiente en el segundo caso. Las líneas 670-730 sólo actúan en el caso de que la potencia del zener, que ha sido introducida, sea igual o menor que la que ha calculado el programa, sugiriendo dos soluciones:

— elevar la beta del transistor serie.

— elevar la potencia de disipación del zener.

FÓRMULAS UTILIZADAS

$Ue = Us/.9$
$If = (1*(Ui/Ue))\ Ii$
$T = 1/f$
$Rc = Us/Is$
$Im = Is/2$
$C = 3*T/Rc$
$Uinv = Ue*SQR(2)$
$Rz = (Ur-Uj)/I2$
$Ib = Is/[beta]$
$Rc = (Ur-Uj)/Ib$

EJEMPLO DE CÁLCULO

DATOS:

Uac(V) = 220
Udc(V) = 12
Isal.(mA) = 500
Fusible(L/R): L
Frec. red(Hz) = 50
U Zener (V) = 12
W Zener (mW) = 1000
[beta] = 20

RESULTADOS:

Fusible(If) = 165 mA
Tensión secundario (Us) = 13.333333 V

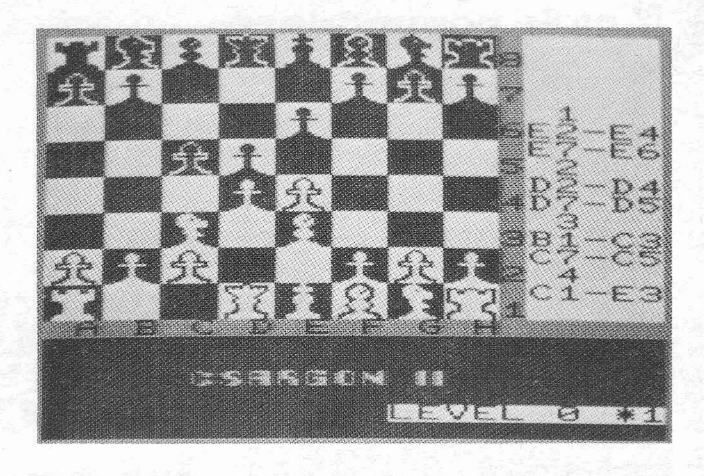
Características diodos:

If = 250mA
Ur = 18.8561808 V
C1 = 2500 microF.
Uc1 = 26.3986532 V
Rz = .0822741701 Kohmios
Wr2 = 571.348404 mW
C2 = 729.268979 microF
Uc2 = 26.3986532 V
Rc = .274247234 Kohmios ■

BANCO DE PRUEBAS

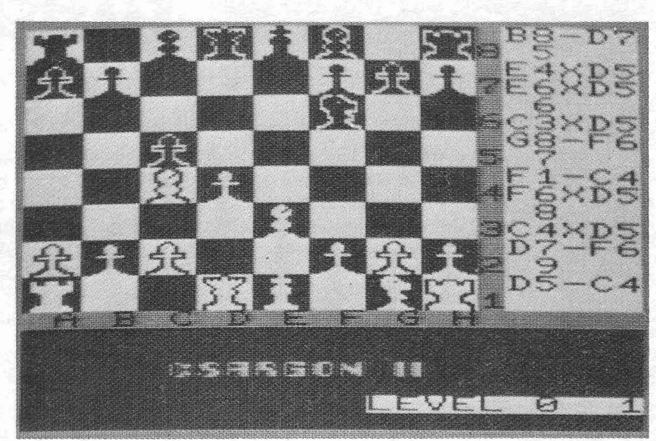
el cartucho de ajedrez del VIC-20 (I)

por **PERE MASATS**

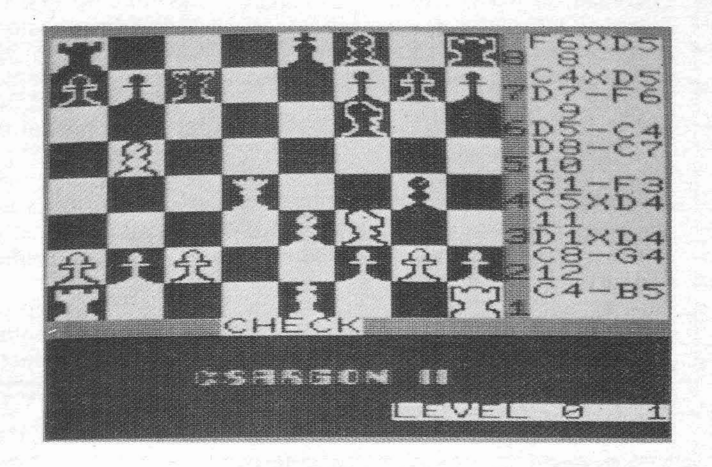


En este número iniciamos una sección que dedicaremos a los diferentes accesorios y programas para los ordenadores personales COMMODORE. En ella iremos dando los detalles más o menos técnicos en la medida en que sean de interés para nuestros lectores y explicaremos su funcionamiento. Para empezar, hoy veremos el cartucho ref. 1919 SARGON II CHESS (AJEDREZ).

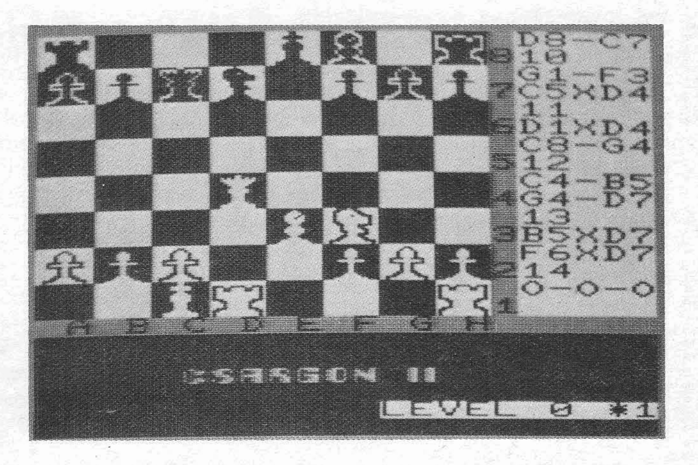
Éste es, sin ninguna duda, el juego más sofisticado y «serio» para el VIC-20. Cuando se da tensión al VIC, después de insertar el cartucho, aparece en pantalla la presentación del juego. Lo primero que debe hacerse es centrar la imagen utilizando las teclas de control de cursor, pudiéndose realizar algunas adaptaciones: 1. En algunos televisores la imagen puede presentar cierta vibración. Para eliminarla hay que pulsar simultáneamente las teclas SHIFT y F7. (En los televisores normales, el pulsar estas teclas puede producir, a su vez, vibraciones.) 2. Se pueden cambiar los colores del juego pulsando F3 y el color del borde con F5.



Para empezar el juego, pulsar F1 y, para terminarlo, RUN/STOP y RESTORE. Al pulsar F1 aparece en la pantalla el mensaje: GAME OR SETUP (G,S): al que se debe responder con una G o una S dependiendo del tipo de juego que queramos disfrutar. Si pulsamos G empezamos un nuevo juego y el VIC se encarga de situar las piezas en la posición inicial; si pulsamos S tenemos la posibilidad de empezar a jugar a partir de una situación dada en el tablero que nosotros podemos definir. (Los jugadores de ajedrez expertos prefieren jugar solamente la última parte de una partida que suele ser la más emocionante.)

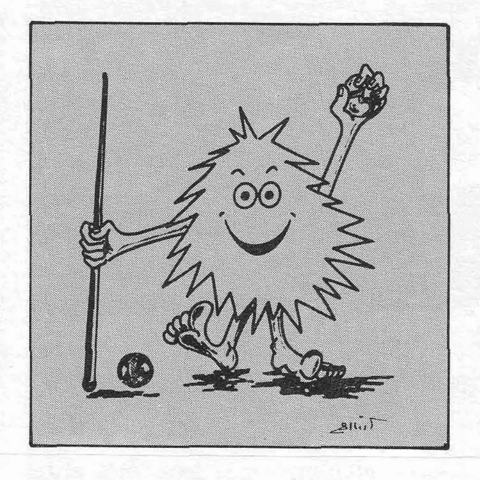


Si se empieza una partida, el VIC preguntará: YOUR COLOR (B,W): a lo que debe responderse con la inicial (en inglés) del color con el que se quiera jugar, o sea, B si queremos jugar con las negras (BLACK) o W si lo queremos hacerlo con las blancas (WHITE). Una vez dado este paso, el VIC preguntará por el nivel de dificultad del juego: LEVEL OF PLAY (0-6): Conforme va subiendo el nivel va creciendo el tiempo de respuesta del ordenador y su «inteligencia». El número de nivel (LEVEL en inglés) representa la cantidad de semi-movimientos futuros del operador que el ordenador tiene en cuenta antes de decidir su jugada. Una vez realizadas estas operaciones, empieza el juego propiamente dicho, distinguiéndose en la pantalla tres partes: arriba y a la izquierda, se sitúa el tablero de ajedrez propiamente dicho, y, en sus bordes, aparecen las letras y los números que nos permitirán realizar los movimientos; en la parte superior derecha, se van registrando los cinco últimos movimientos realizados en notación de ajedrez estándard; en la parte baja de la pantalla aparecen los mensajes del juego.



(continuará)





simulación física del choque elástico: juego de billar

por

DAVID CERVIGÓN FERNÁNDEZ
I.N.B. "Cardenal Herrera Oria"
COU-A

A David Cervigón Fernández le conocimos en Madrid durante la última edición de SIMO en noviembre pasado. Como habíamos proferido estentóreas voces (desde CLUB COMMODORE y otros lugares), clamando por las aplicaciones educativas del VIC, un compañero de Madrid nos puso en contacto (¡gracias!) con el Instituto Experimental de Bachillerato "Cardenal Herrera Oria" que obtuvo cinco de los diez premios del Torneo Escolar de Programación 82. El trabajo que presentamos aquí ganó el tercer premio de dicho concurso. ¡Felicitaciones y gracias por el artículo!

El fenómeno del choque o colisión se produce cuando dos partículas o sistemas de partículas se aproximan entre sí alterando su movimiento y produciendo un intercambio en su cantidad de movimiento (P) y energía (T).

1) DESCRIPCIÓN DEL PROBLEMA

Nuestro estudio se circunscribe al caso en que realmente existe contacto físico entre las partículas (choque de dos bolas de billar) y en donde se produce una dispersión, es decir, las partículas resultantes del fenómeno del choque son las mismas que las iniciales. La colisión se producirá en una única dirección. Será completamente elástica, frontal, y respetará las leyes físicas de la reflexión.

El volumen de las bolas será el mismo no siendo así su masa, debido a que el jugador podrá variarlas cuando lo desee. Asimismo podrá variar la velocidad inicial de la bola y, estrechamente ligado, el espacio total que deberá recorrer. Todas estas magnitudes predeterminarán los resultados del choque, fenómeno producido por el sistema de ecuaciones siguiente:

(1): M1*V1+M2*V2=M1*V1'+M2*V2'

(2): 1/2M1*V1^2+1/2M2*V2^2=1/2M1*V1'^2+1/2M2+V2'^2

Donde la ecuación (1) es el principio de conservación de la cantidad de movimiento (P), y la ecuación (2) es el de conservación de la energía cinética (T). Y V1 y V2 son — respectivamente — las velocidades de la bola 1 y de la bola 2 antes del choque y V1', V2' después del choque.

Las masas M1 y M2 se mantendrán constantes en todo momento. El estado físico de la bola 2 (bola amarilla) antes del choque es el reposo (V=0) y, por limitaciones del ordenador, tras la colisión desaparecerá. Por lo tanto, el estudio científico se restringe al movimiento de una sola bola (bola roja), la cual partirá tras el choque con una velocidad V1' igual a:

V1'=((M1-M2)*V1)/(M1+M2)

2) DESARROLLO DEL TRABAJO

Atendiendo a los medios materiales utilizados, el programa ha sido realizado en dos fases:

La primera, en el centro de enseñanza antes citado, con un PET 2001 (COMMODORE) sin ningún tipo de periféricos. Y la segunda, con un ordenador VIC-20 (COMMODORE) propio. En este caso se ha utilizado un cartucho de ampliación de memoria 16 KB y una impresora COMMODORE que han sido proporcionados, desinteresadamente, por la casa de microsistemas ACCORD SOFT.

He consultado los manuales de ambas máquinas junto con revistas especializadas españolas (EL ORDENADOR PERSONAL), de aparición muy reciente, y extranjeras (YOUR COMPUTER, publicación inglesa) para resolver los problemas de programación; y el libro «FÍSICA COU», autores E. Lowy y otros, para resolver los problemas de simulación científica. Debo destacar, por último, la gran ayuda prestada en ambos campos por profesores de los seminarios de Matemáticas y Física del Instituto «Cardenal Herrera Oria», que han hecho posible, en gran medida, la presentación de este trabajo.

3) EXPLICACIÓN DEL PROGRAMA

Para comprender la estructura del programa mostramos a continuación un organigrama que, a grandes rasgos, pretende ofrecer una idea clara y global de lo que es el programa.

La partida en sí es una subrutina que se efectúa para cada jugador tantas veces como sea necesario. Está formada por varias partes que analizan por separado cada una de las situaciones que se van presentando. El jugador puede mover el taco en ocho direcciones distintas y jugará con la bola roja. El impulso de ésta, que se consigue pulsando la tecla G, tiene cinco grados; cuanto más separado esté el taco de la bola, mayor velocidad inicial le imprimirá y, por lo tanto, mayor espacio recorrerá. El choque, bien con una banda o con otra bola, decelerará a la bola según las condiciones existentes. También cada espacio que recorra sufrirá una deceleración, aunque ésta será mínima.

La velocidad de la bola se consigue mediante bucles retardadores situados en cada uno de los tres tipos de movimiento (vertical, horizontal y oblicuo).

Al principio del juego, las bolas siempre saldrán en una determinada posición pero, una vez borradas de la pantalla debido a la colisión con la bola roja, serán situadas aleatoriamente en la próxima jugada. Todo esto, el movimiento del taco, los choques, etc., llevan aparejados ciertos efectos especiales, como son diferentes colores y sonidos para cada caso, consiguiendo de esta manera una mayor vistosidad del juego.

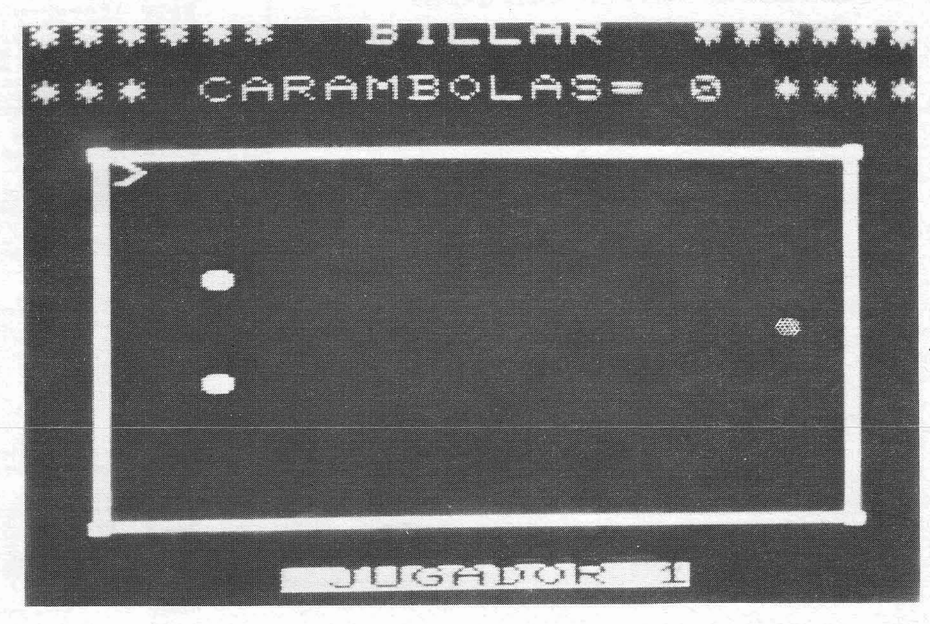


Fig. 1

4) PRESENTACIÓN GRÁFICA

En primer lugar, el ordenador preguntará si se desea conocer las reglas del juego. En caso afirmativo, las im-

(continúa en la pág. 13)

```
1 REM BILLAR
5 REM POR DAVID CERVIGON FERNANDEZ
6 M1=250:M2=250
7 POKE36879,15:POKE36878,15
9 PRINT"________________________________"
11 PRINT"    SIMULACION FISICA   ":PRIN
T
12 PRINT" DEL CHOQUE ELASTICO   "
13 PRINT"                         "
14 PRINT"                         "
15 PRINT"                         "
16 PRINT"                         "
17 PRINT"                         "
18 PRINT"                         "
19 PRINT"                         "
20 PRINT"               POR"
21 PRINT" DAVID CERVIGON FEDZ. "
22 FORLP=1TO100
25 POKE36876,INT(RND(TI)*128)+128
26 FORPL=1TO10
27 NEXTPL
28 NEXTLP
30 POKE36876,0
35 PRINT"DESEA VER LAS REGLAS?"
40 GETN$:IFN$=""THEN40
45 IFN$="N"THENPOKE36876,200:POKE36876,0
:GOTO2000
50 IFN$<>"N"ANDN$<>"S"THEN40
55 POKE36876,220:POKE36876,0
1000 REM REGLAS DEL JUEGO
1010 PRINT"              REGLAS"
1020 PRINT"            ________"
1025 POKE36876,135:FORT=1TO500:NEXTT:POK
E36876,0
1030 PRINT"LA FINALIDAD DEL JUEGO"
1035 POKE36876,147:FORT=1TO500:NEXTT:POK
E36876,0
1040 PRINT"ES HACER DESAPARECER  "
1045 POKE36876,159:FORT=1TO500:NEXTT:POK
E36876,0
1050 PRINT"LAS BOLAS AMARILLAS.  "
1055 POKE36876,163:FORT=1TO1000:NEXTT:PO
KE36876,0
1060 PRINT" PARA MOVER EL TACO  :  "
1065 POKE36876,175:FORT=1TO500:NEXTT:POK
E36876,0
1070 PRINT"T-ARRIBA,B-ABAJO,F-IZQ"
1075 POKE36876,183:FORT=1TO500:NEXTT:POK
E36876,0
1080 PRINT"H-DCHA.N-R-Y-V-OBLICUO"
1085 POKE36876,191:FORT=1TO500:NEXTT:POK
E36876,0
1090 PRINT"G-IMPULSO.MAX5ESPACIOS"
1095 POKE36876,195:FORT=1TO1500:NEXTT:PO
KE36876,0
1100 PRINT" UV.JUEGA CON LA BOLA "
1105 POKE36876,201:FORT=1TO500:NEXTT:POK
E36876,0
1110 PRINT"ROJA.DESEA VARIAR LAS "
1115 POKE36876,207:FORT=1TO500:NEXTT:POK
E36876,0
1120 PRINT"CONDICIONES FISICAS   "
1125 POKE36876,209:FORT=1TO500:NEXTT:POK
E36876,0
1128 PRINT"DEL JUEGO (S 0 N) ? "
1130 GETH$:IFH$=""THEN1130
1140 IFH$="N"THENPOKE36876,250:POKE36876
,0:GOTO2000
1150 IFH$<>"N"ANDH$<>"S"THEN1130
1155 POKE36876,250:POKE36876,0
1160 REM*******************************
***
1200 REM FUNDAMENTO FISICO
1210 PRINT"***SIMULACION FISICA***"
1230 PRINT"            DATOS":PRINT:PRINT
1240 PRINT"250GR.MASA STANDAR DE "
1250 PRINT"LAS BOLAS DE BILLAR.   ":PRINT
1260 PRINT"-BOLA ROJA :"
1270 INPUT"MASA EN GRAMOS "
:M1:PRINT:PRINT
1280 IFM1<=0THEN1270
1285 POKE36876,250:POKE36876,0
1290 PRINT"-BOLAS AMARILLAS :":PRINT
1300 INPUT"MASA EN GRA
MOS ":M2:PRINT
1310 IFM2<=0THEN1300
1315 POKE36876,250:POKE36876,0
1320 PRINT"OK":FORT=1TO1500:NEXTT
1350 REM*******************************
***
2000 REM PROLOGO DE LA PARTIDA
2010 K=4206:W=62
2020 REM SITUACION INICIALDE LAS BOLAS
2030 BR=4356:B1=4298:B2=4386
2050 REM DATOS PARA EL JUEGO
2060 PRINT"*** BILLAR ESPANOL ***":PRI
NT
2070 INPUT"NUMERO DE JUGADORES";J:P
RINT
2080 IFJ<=0THEN2070
2085 POKE36876,220:POKE36876,0
2090 PRINT"LA PARTIDA ACABARA  :":PRINT
2100 PRINT"-TIEMPO (1)":PRINT
2110 PRINT"-N.DE CARAMBOLAS (2)":PRINT
2120 PRINT"-AMBOS A LA VEZ (3)":PRINT
2130 PRINT" TECLEE 1,2 O 3":PRINT
2140 GETA$:IFA$=""THEN2140
2150 IFA$<>"1"ANDA$<>"2"ANDA$<>"3"THEN21
40
2155 POKE36876,250:POKE36876,0
2160 IFA$="2"THEN2190
2170 INPUT"ESCRIBALO(00
0000)";T$:PRINT
2175 LO=VAL(T$)
2176 IFLO<=0THEN2170
2177 POKE36876,250:POKE36876,0
2180 IFA$="1"THEN2200
2185 POKE36876,250:POKE36876,0
2190 INPUT"CUANTAS C
ARAMBOLAS";C:PRINT
```

(continúa en la pág. siguiente)

simulación física del choque elástico: juego de billar

(viene de la pág. anterior)

```
2195 IFC<=0THEN2190
2196 POKE36876,250:POKE36876,0
2200 PRINT"OK"
2210 REM DIMENSION DE LA MATRIZ DE LOS J
UGADORES
2220 DIMC(2,J)
2230 FORJU=1TOJ
2240 A(JU,1)=JU
2250 NEXTJU
2260 FORT=1TO1000:NEXTT:TI$="000000"
2270 PRINT"****** BILLAR ******"
2290 GOSUB7000
2300 REM JUEGO
2310 FORJU=1TOJ
2320 PRINT"
 JUGADOR";JU;"":CA=0:CO=0
2330 PRINT"*** CARAMBOLAS=";A(JU,2);
"****":GOSUB3000
2340 PRINT""
2350 K=4206:W=62:V=0:R=0:E=0:S=0:Y=0:X=0
:D=0
2360 I=0:B=0:H=0:DA=0:DR=0:CO=0
2370 IFPEEK(B1)<>81ANDPEEK(B2)=81THENGOS
UB4000:GOTO2490
2375 IFPEEK(B2)<>81ANDPEEK(B1)=81THENGOS
UB4010:GOTO2490
2380 IFPEEK(B1)<>81ANDPEEK(B2)<>81THENGO
SUB4000:GOSUB4010:GOTO2400
2390 GOTO2490
2400 REM PUNTOS
2410 CA=CA+1:A(JU,2)=CA
2420 PRINT"*** CARAMBOLAS=";A(JU,2);
"****"
2430 REM EPILOGO DE LA PARTIDA
2440 IFA$="2"THEN2470
2450 IFTI$>=T$THENPRINT"
FIN":GOTO2510
2460 IFA$="1"THEN2330
2470 IFCA>=CTHENPRINT"
FIN":GOTO2510
2480 GOTO2330
2490 NEXTJU
2500 GOTO2290
2505 REM************************************
***
2510 REM GANADOR
2520 MAX=A(1,2):LMAX=A(1,1):POKEB1,81:PO
KEB2,81
2530 FORJU=2TOJ
2540 IFA(JU,2)>MAXTHENMAX=A(JU,2):LMAX=A
(JU,1)
2550 NEXTJU
2560 PRINT"GANADOR: JUGADOR"
;LMAX:PRINT
2570 PRINT"CON ";MAX;" CARAMBOLAS":FOR
T=1TO5000:NEXTT
2575 REM************************************
***
2580 REM FIN
2590 PRINT"DESEA
SEGUIR JUGANDO?"
2595 GETM$:IFM$=""THEN2595
2620 IFM$="S"THENRUN
2630 IFM$="N"THENEND
2640 IFM$<>"S"ANDM$="N"THEN2595
2650 END
3000 REM************************************
***
3005 REM SUBRUTINA:JUEGO
3010 REM REPRESENTACION GRAFICA:BOLAS Y
TACO
3015 IFB1=BRTHENGOSUB4000
3016 IFB2=BRTHENGOSUB4010
3020 POKEBR+33792,2:POKEB1+33792,7:POKEB
2+33792,7:POKEK,W:POKEBR,81:POKEB1,81:PO
KEB2,81
3035 FORT=1TO750:NEXTT
3040 POKEK,32:K=K+2
3045 P=PEEK(K)
3050 IFP=108ORP=124ORP=126ORP=118ORP=123
ORP=117ORP=121ORP=120THENK=M
3085 POKE36876,220:POKEM,32:POKEK+33792,
1:POKEK,W:POKE36876,0
3090 GETB$:IFB$=""THEN3090
3100 IFB$="R"THENK=K-23:W=60:M=K+23
3110 IFB$="T"THENK=K-22:W=62:M=K+22
3120 IFB$="Y"THENK=K-21:W=62:M=K+21
3130 IFB$="F"THENK=K-1:W=60:M=K+1
3140 IFB$="H"THENK=K+1:W=62:M=K-1
3150 IFB$="V"THENK=K+21:W=60:M=K-21
3160 IFB$="B"THENK=K+22:W=60:M=K-22
3170 IFB$="N"THENK=K+23:W=62:M=K-23
3180 IFB$="G"THENPOKEK,32:GOTO3200
3190 POKEBR,81:POKEB1,81:POKEB2,81:POKEB
1+33792,7:POKEB2+33792,7:POKEBR+33792,2:
GOTO3045
3195 REM************************************
***
3200 REM PROPIEDADES DEL MOVIMIENTO
3205 POKEB1+33792,7:POKEB1,81:POKEB2+337
92,7:POKEB2,81
3210 FORY=1TO5
3220 IFK=BR-23*YTHENGOSUB5500:GOTO3865
3230 IFK=BR-21*YTHENGOSUB5500:GOTO3805
3240 IFK=BR-22*YTHENGOSUB5500:GOTO3630
3250 IFK=BR-1*YTHENGOSUB5500:GOTO3430
3260 IFK=BR+1*YTHENGOSUB5500:GOTO3410
3265 IFK=BR+21*YTHENGOSUB5500:GOTO3825
3270 IFK=BR+22*YTHENGOSUB5500:GOTO3610
3280 IFK=BR+23*YTHENGOSUB5500:GOTO3845
3290 NEXTY
3300 GOTO3045
3310 REM************************************
***
3400 REM MOV. HORIZONTAL
3410 REM DIRECCION IZQUIERDA
3420 IFPEEK(BR-1)=118THEN3445
3425 U=-1:GOTO3450
3430 REM DIRECCION DERECHA
3435 IFPEEK(BR+1)=117THEN3425
3445 U=1
3450 X=0:D=0
3460 POKEBR+D,32:POKEBR+D+33792,0
3470 D=D+U:P=PEEK(BR+D+1):E=PEEK(BR+D-1)
3480 IFP=117ORE=118THENU=-U:POKE36876,22
0:V=V+4:X=X+4
3490 IFPEEK(BR+D)=81THENPOKEBR+D,32:BC=B
R+D:GOSUB6000:POKEBR+D+33792,0:GOTO3480
3500 X=X+1:V=V+1:POKEBR+D+33792,2:POKEBR
+D,81
3510 FORT=1TOV:NEXTT:POKE36876,0
3520 IFX>=STHEN3550
3530 GOTO3460
3550 BR=BR+D:RETURN
3555 REM************************************
***
3600 REM MOV. VERTICAL
3610 REM DIRECCION ARRIBA
3620 IFPEEK(BR-22)=121THEN3645
3625 Q=-1:GOTO3650
3630 REM DIRECCION ABAJO
3640 IFPEEK(BR+22)=120THEN3625
3645 Q=1
3650 X=0:O=0
3660 POKEBR+22*O,32:POKEBR+22*O+33792,0
3670 O=O+Q:P=PEEK(BR+22*O+22):E=PEEK(BR+
22*O-22)
3680 IFP=120ORE=121THENQ=-Q:POKE36876,23
0:V=V+5:X=X+8
3690 IFPEEK(BR+22*O)=81THENPOKEBR+22*O,3
2:BC=BR+22*O:GOSUB6030:POKEBR+22*O+33792
,0:GOTO3680
3700 X=X+1:V=V+1:POKEBR+22*O+33792,2:POK
EBR+22*O,81
3710 FORT=1TOV:NEXTT:POKE36876,0
3720 IFX>=STHEN3750
3740 GOTO3660
3750 BR=BR+22*O:RETURN
3760 REM************************************
***
3800 REM MOV.OBLICUO
3805 REM DIRECCION SUR-OESTE
3810 IFPEEK(BR+22)=120THEN3850
3815 IFPEEK(BR-1)=118THEN3870
3820 DA=-1:DR=1:GOTO3890
3825 REM DIRECCION NORTE-ESTE
3830 IFPEEK(BR-22)=121THEN3870
3835 IFPEEK(BR+1)=117THEN3850
3840 DA=1:DR=-1:GOTO3890
3845 REM DIRECCION NORTE-OESTE
3850 IFPEEK(BR-22)=121THEN3810
3855 IFPEEK(BR-1)=118THEN3830
3860 DA=-1:DR=-1:GOTO3890
3865 REM DIRECCION SUR-ESTE
3870 IFPEEK(BR+22)=120THEN3830
3875 IFPEEK(BR+1)=117THEN3810
3880 DA=1:DR=1
3890 A=0:R=0:X=0
3900 POKEBR+A+22*R,32:POKEBR+A+22*R+3379
2,0
3910 A=A+DA:P=PEEK(BR+A+22*R+1):E=PEEK(B
R+A+22*R-1)
3915 IFP=117ORE=118THENDA=-DA:POKE36876,
220:V=V+3:X=X+4
3920 IFPEEK(BR+A+22*R)=81THENPOKEBR+A+22
*R,32:BC=BR+A+22*R:GOSUB6000:POKEBR+A+22
*R+33792,2:GOTO3915
3925 R=R+DR:P=PEEK(BR+A+22*R+22):E=PEEK(
BR+A+22*R-22)
3930 IFP=120ORE=121THENDR=-DR:POKE36876,
230:V=V+3:X=X+4
3935 IFPEEK(BR+A+22*R)=81THENPOKEBR+A+22
*R,32:BC=BR+A+22*R:GOSUB6030:POKEBR+A+22
*R+33792,2:GOTO3930
3940 X=X+1:V=V+1:POKEBR+A+22*R+33792,2:P
OKEBR+A+22*R,81
3945 FORT=1TOV:NEXTT:POKE36876,0
3950 IFX>=STHEN3970
3960 GOTO3900
3970 BR=BR+A+22*R:RETURN
3980 REM************************************
***
3990 REM COORDENADAS DE LAS BOLAS
4000 B1=4208+INT(RND(TI)*18)+22*INT(RND(
TI)*14):GOTO4020
4010 B2=4208+INT(RND(TI)*18)+22*INT(RND(
TI)*14)
4020 IFB1=B2ORB1=BRORB2=BRTHEN4000
4030 RETURN
5500 REM************************************
***
5510 K=4206:W=62:POKEK,W:POKEBR,32
5520 REM VELOCIDAD Y ESPACIO
5530 V=(60/Y)-5
5540 S=50*Y-20
5550 RETURN
```

```
6000 REM*****************************
***
6010 REM COLISION
6020 IFM1<=M2THENU=-U:DA=-DA:GOTO6050
6030 IFM1<=M2THENQ=-Q:DR=-DR
6050 VC=((M1-M2)*V)/(M1+M2)
6055 POKE36876,240:X=X+10
6060 POKEBC-23+33792,7:POKEBC-23,126:POK
EBC+23,108:POKEBC+23+33792,7
6065 POKEBC-21,124:POKEBC-21+33792,7:POK
EBC+21,123:POKEBC+21+33792,7
6095 KP=50
6096 IFVC>100THENKP=5000
6098 IFVC=0THENVB=1:GOTO6120
6099 IFVC<0THENVC=-VC
6100 VB=KP/VC
6105 IFVB<0THENVB=-VB
6110 VB=INT(VB)
6120 V=V+VB
6130 POKEBC-23,32:POKEBC+23,32:POKEBC-21
,32:POKEBC+21,32
6135 POKEBC-23+33792,0:POKEBC-21+33792,0
:POKEBC+23+33792,0:POKEBC+21+33792,0
6138 IFBC<>B1ANDCO=0THENPOKEB1+33792,7:P
OKEB1,81:CO=1
6139 IFBC<>B2ANDCO=0THENPOKEB2+33792,7:P
OKEB2,81:CO=1
6140 POKE36876,0:GOSUB7000:RETURN
7000 REM MESA
7010 FORI=4203TO4186STEP-1:POKEI,121:POK
EI+33792,1:NEXTI:POKE4185,108
7020 FORI=4207TO4493STEP22:POKEI,118:POK
EI+33792,1:NEXTI:POKE4515,124
7030 FORI=4516TO4533:POKEI,120:POKEI+337
92,1:NEXTI:POKE4534,126
7040 FORI=4512TO4196STEP-22:POKEI,117:PO
KEI+33792,1:NEXTI:POKE4204,123
7050 RETURN

READY.
```

primirá en la pantalla, dando la posibilidad, mediante una nueva pregunta, a que varíe las condiciones físicas en que se desarrollará el juego. En caso negativo, bien sea a la primera o a la segunda cuestión, se imprimirán automáticamente en pantalla dos preguntas: la primera hace referencia al número de jugadores que desean participar. En este caso no hay restricciones, pudiendo jugar todos aquellos que lo deseen. Y la segunda, cuáles van a ser los límites de la partida, es decir, si la partida finalizará cuando transcurra un cierto tiempo (esta modalidad está dirigida al caso de que sólo juegue una persona, a modo de entrenamiento), o cuando se cumplan un cierto número de carambolas (modalidad dirigida al caso en que haya numerosos jugadores). También se puede elegir las dos modalidades anteriores a la vez. En el segundo caso, el jugador ganador será aquel que consiga, el primero, alcanzar el número de carambolas preestablecido. Estas dos preguntas se deberán responder obligatoriamente, ya que son datos esenciales para el juego.

Una vez cumplidos todos estos requisitos, dará comienzo el juego, en la parte superior de la pantalla aparecerá «BILLAR» y una línea debajo el puntuador. En el centro de la pantalla la mesa de billar con tres bolas, dos amarillas y una roja, situadas inicialmente en un lugar fijo, que por supuesto permite lograr hacer una carambola.

En la esquina superior izquierda, fuera de la mesa, aparecerá el taco, donde tras permanecer unos segundos, atravesará la mesa. A partir de este instante el jugador comenzará su partida, moviendo el taco como lo desee, con la única restricción de que no podrá sacarlo de la mesa.

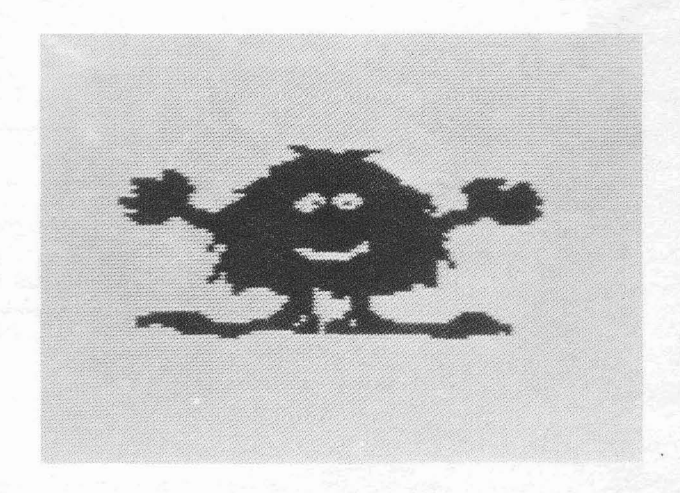
En resumen, en la figura 1 se da la situación gráfica antes de que comience el juego.

5) CONCLUSIONES

El programa, en líneas generales, ha sido concebido para que no se puedan dar situaciones imprevistas cuyos desenlaces sean indeterminables, así como para que pueda ser utilizado por cualquier persona, ya que no se necesita ningún tipo de conocimientos previos de programación ni de manejo de ordenadores y su utilización es muy simple.

He intentado, en este programa, sintetizar el fundamento físico o científico existente en el billar, con sus características de juego como la habilidad, el ingenio, etc., y todo ello en el marco de las limitaciones propias del ordenador, tales como la rapidez de lectura, resolución, etc. Que esta síntesis se haya logrado o no, lo deberán decidir ustedes. ■

COLABORACIONES



BUG sale en Televisión

por ERNESTO MARTÍNEZ DE CARVAJAL HEDRICH

No creemos necesario presentar a ERNESTO dado que una colaboración suya se publicó en el número cuatro de nuestra Revista. Como se puede apreciar en este artículo nuestro autor se las ha arreglado para que un personaje de nuestra Revista (por cierto: el malo) aparezca en televisión, lo cual no deja de ser un honor... ¿no?

Apreciados conVICtos:

En mi anterior colaboración os describí un programa (misioneros y caníbales) que utilizaba una serie de caracteres especiales que representaban los personajes y demás elementos del juego. En dicha ocasión no os detallé más este apartado por no ser la intención del artículo y porque es un tema que requiere uno monográfico. En esta ocasión quisiera introduciros a los que no lo estáis, en esta faceta apasionante del VIC: la generación de caracteres definidos por el usuario.

Una de las innovaciones que presenta el VIC es la posibilidad de definirle caracteres y utilizarlos en programas BASIC. Las aplicaciones de esto son diversas. Se puede sustituir los caracteres del VIC por aquellos propios de una lengua, como el japonés, el chino o cualquier otro. Se puede crear una tabla de caracteres electrónicos, matemáticos, etc. Y se pueden crear las más extrañas criaturas de ciencia-ficción.

En este artículo explico (o al menos intento) la manera de generar este tipo de caracteres, a la vez que os presento un programa que podéis utilizar como subrutina de otros para realizar esta función. Y para ilustrarlo, qué mejor que utilizar a nuestra mascota BUG que, por una vez, nos ayudará en lugar de hacer travesuras.

Ante todo, el necesario y consabido repaso de la teoría:

Se puede considerar que la pantalla del VIC (el recuadro interior que aparece en la pantalla) está formado por puntos elementales. Cada uno de estos puntos puede estar encendido o apagado, entendiéndose por apagado el color del fondo de la pantalla (especificado en la dirección 36879), y por encendido de un determinado color (contenido en la dirección 38400). Estos puntos forman un entramado de 184 líneas por 176 columnas. El controlador de video monta la imagen línea a línea y punto a punto, por lo que ha de conocer el estado, encendido o apagado, de cada uno de ellos. Este estado se podría codificar con un BIT (0 = apagado, 1 = encendido), lo cual representaría una información de 176*184 = 32.384 BITS = 4048 octetos = 4K, lo cual sería excesivo frente a las 5,5 K totales del VIC en su versión base. Ha sido necesario buscar una solución más económica. Para ello se agrupan los puntos elementales en rectángulos de ocho líneas por ocho columnas, llamados *mallas*. Estas mallas se reparten en la pantalla en filas y columnas. Es preciso distinguir entre filas y columnas, que podemos llamar *macroscópicas*, de las elementales. La pantalla está formada por 23 filas de 22 columnas (macroscópicas). La combinación de caracteres que se puede dibujar dentro de una malla es bastante limitada: 2*64 = 256, ya que hay 64 puntos (8*8) y cada uno puede tener dos estados.

La forma en que se las ingenia el VIC para visualizar caracteres es la siguiente:

En una zona de RAM de número de octetos igual al número de mallas de la pantalla (506 para el VIC), se introduce el código ASCII del carácter que ha de visualizar en su correspondiente malla. Si llamamos E a la dirección de origen de esta zona (7680), la dirección en memoria del octeto correspondiente a la malla de coordenadas XM,YM será:

$$Q = E + 22*YM + XM$$

Una segunda memoria, en ROM o RAM, *contiene la descripción correspondiente* a cada código. Esta memoria se llama generador de caracteres. Hay un octeto por cada carácter. Si llamamos G al origen de esta memoria (32768 en el VIC al encenderlo), en G+8*R+L se encuentra el octeto que es la imagen de la línea elemental L, necesaria para dibujar el carácter R. Los ocho bits de un octeto indican el estado de cada uno de los ocho puntos de una línea dentro de una malla. Así, por ejemplo, la forma en que el VIC guarda la letra A se muestra en la tabla de la página siguiente.

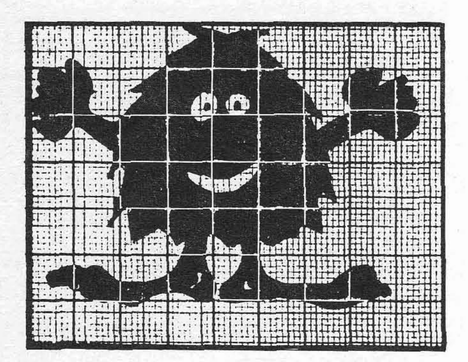
Se pueden comprobar estos valores ejecutando el siguiente programa:

```
10 G=32768 : REM dirección inicio
   generador
20 INPUT " carácter a decodificar";R
30 FOR L=O TO 7
40 PRINT PEEK (G+8*R+L)
50 NEXT L
60 PRINT:PRINT
70 GOTO 20
```

Esta segunda memoria se puede trasladar a una zona RAM y, lo que es

mejor, modificarla a nuestra voluntad e instruir al VIC para que utilice esta nueva tabla para la generación de los caracteres.

Y llegados a este punto es el momento de empezar con nuestro amiguete. Evidentemente no vamos a intentar dibujarlo utilizando un solo carácter, ya que esto nos daría una resolución muy pobre (64 puntos). La idea es trocear a BUG en unidades representables en una malla, y después unirlas en la pantalla como si fuese un puzzle. Para ello dibujemos a BUG sobre un papel milimetrado (el dibujo escogido es el que aparece en el número uno de nuestra Revista para la presentación de la mascota), de forma que la parte izquierda y superior coincida con el inicio de un centímetro. A continuación dividamos el dibujo en cuadrados de ocho milímetros de lado. Cada cuadrado de éstos representa una malla o un carácter, y cada uno de los milímetros cuadrados interiores un bit. El resto es fácil aunque laborioso: hay que calcular el valor de cada uno de los ocho octetos de cada carácter e introducirlos en el lugar que queremos modificar.



FILA	MALLA	DIRECCIÓN	ESTADO BITS	VALOR OCTETO
1	**	32776	00011000	2^4+2^3 =24
2	* *	32777	00100100	2^5+2^2 = 36
3	* *	32778	01000010	2^6+2^1 = 66
4	******	32779	01111110	2^6+2^5+2^4+2^3 +2^2+2^1 = 126
5	* *	32780	01000010	2^6+2^1 = 66
6	* *	32781	01000010	2^6+2^1 = 66
7	* *	32782	01000010	2^6+2^1 = 66
8		32782	00000000	= 0

DESCRIPCIÓN DEL PROGRAMA

La línea 20 del programa inicializa los valores de las variables:

G1 = Inicio generador de caracteres en ROM.

G2 = Inicio generador de caracteres en RAM (puede tener otros).

La línea 60 traslada la tabla de caracteres de la ROM a la RAM.

Las líneas 1000-1124 son los valores de cada uno de los ocho octetos de cada carácter. El primer valor de cada data es el código (según tabla 1 de pokes) del carácter a modificar.

La línea 1510 realiza la modificación de los caracteres según el valor de los datas.

La línea 1600 protege la tabla de caracteres en RAM.

La línea 2001 modifica el orden de la tabla de generación de caracteres. (Para restituir el valor original pulsar RUN/STOP y restore.)

Las líneas 2005-2070 pintan el dibujo en la pantalla. ■

```
1 PRINT"■":POKE36879,25
10 REM GENCHAR
20 G1=32768:G2=7168:IP=7680
60 FORI=0TO512:K=PEEK(G1+I):POKE(G2+I),K
:NEXTI
1000 DATA 0,0,0,0,0,0,0,1,1
1002 DATA 1,0,0,0,0,0,0,128,192
1004 DATA 2,0,0,0,0,0,0,0,0
1006 DATA 3,28,15,7,15,63,255,255,255
1008 DATA 4,0,207,255,254,255,255,255,25
5
1010 DATA 5,0,192,0,0,128,192,224,240
1012 DATA 6,0,0,0,0,0,0,0,0
1014 DATA 7,0,0,0,0,0,0,0,0
1016 DATA 8,0,0,0,0,0,0,0,0
1018 DATA 9,61,63,31,63,31,255,255,127
1020 DATA 10,224,224,240,240,224,224,224
,240
1022 DATA 11,3,7,15,63,15,31,63,127
1024 DATA 12,255,255,255,255,252,248,241
,240
1026 DATA 13,255,255,255,255,227,65,73,2
27
1028 DATA 14,248,248,254,255,255,255,255
,255
1030 DATA 15,0,0,1,1,192,192,192,224
1032 DATA 16,124,252,249,255,255,255,255
,255
1034 DATA 17,0,192,192,224,224,240,240,2
24
1036 DATA 18,31,127,63,30,0,0,0,0
1038 DATA 19,252,255,63,31,15,3,1,1
1040 DATA 20,255,255,255,255,255,255,255
,255
1042 DATA 21,249,255,255,255,255,255,255
,255
1044 DATA 22,255,255,255,255,255,255,255
,255
1046 DATA 23,255,255,255,255,255,255,255
,255
1048 DATA 24,167,207,255,255,252,248,240
,248
1050 DATA 25,255,255,207,135,0,0,0,0
1052 DATA 26,240,240,224,192,0,0,0,0
1054 DATA 27,0,0,0,0,0,0,0,0
1056 DATA 28,1,2,2,0,0,0,1,1
1058 DATA 29,255,127,127,255,255,255,255
,255
1060 DATA 30,255;255,241,248,252,255,255
,255
1062 DATA 31,254,252,248,3,7,255,255,255
1064 DATA 32,255,255,255,255,255,255,255
,255
1066 DATA 33,240,240,248,248,248,228,228
,224
1068 DATA 34,0,0,0,0,0,0,0,0
1070 DATA 35,0,0,0,0,0,0,0,0
1072 DATA 36,0,0,0,0,0,0,0,0
1074 DATA 37,1,3,2,0,0,0,0,0
1076 DATA 38,127,127,127,63,48,32,0,0
1078 DATA 39,255,255,255,255,254,126,30,
30
1080 DATA 40,255,255,255,255,255,63,55,7
1082 DATA 41,255,255,255,254,254,240,0,0
1084 DATA 42,224,224,224,96,0,0,0,0
1086 DATA 43,0,0,0,0,0,0,0,0
1088 DATA 44,0,0,0,0,0,0,0,0
1090 DATA 45,0,0,7,31,31,25,0,0
1092 DATA 46,0,0,248,252,254,255,127,63
1094 DATA 47,0,0,0,0,7,255,255,248
1096 DATA 48,62,62,126,251,251,247,255,1
5
1098 DATA 49,7,7,15,15,59,123,127,127
1100 DATA 50,0,0,128,192,240,255,255,255
1102 DATA 51,0,0,1,3,7,255,255,255
1104 DATA 52,0,0,224,240,248,254,254,250
1106 DATA 53,0,0,0,0,0,0,0,0
1108 DATA 54,0,0,0,0,0,0,0,0
1110 DATA 55,0,0,0,0,0,0,0,0
1112 DATA 56,224,0,0,0,0,0,0,0
1114 DATA 57,0,0,0,0,0,0,0,0
1116 DATA 58,0,0,0,0,0,0,0,0
1118 DATA 59,0,0,0,0,0,0,0,0
1120 DATA 60,63,0,0,0,0,0,0,0
1122 DATA 61,0,0,0,0,0,0,0,0
1124 DATA 62,0,0,0,0,0,0,0,0
1510 FORI=0TO62:READC:PRINTC:FORJ=0TO7:R
EADK:POKEG2+(C*8)+J,K:NEXTJ:NEXTI
1600 POKE 52,28:POKE 56,28
2000 PRINT "■■■■■■■■■■■■■■■■■";
2001 POKE 36869,255
2005 K=63
2010 FOR I=1 TO 7
2020 FOR J=1TO 9
2030 K=K+1
2035 IF K>95 THEN K=K-64
2040 PRINT CHR$(K);
2050 NEXT J
2060 PRINT:PRINT"■■■■■■■■";
2070 NEXT I
2080 GETA$:IFA$=""THEN2080

READY.
```

CLUBS DE USUARIOS

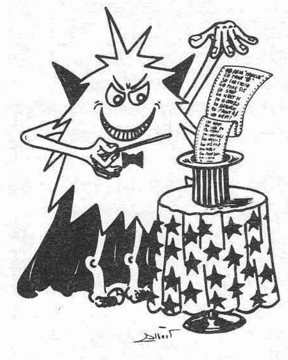
el primer Club de Usuarios del VIC-20 en Barcelona ya está en marcha

Confirmando las noticias anticipadas en ediciones anteriores, ha quedado constituido el primer Club de Usuarios del VIC-20 en Barcelona. La primera reunión se celebró, de acuerdo con el programa previsto, en «Sistema» — Balmes, 434 — Barcelona, Surgieron diversos proyectos que se espera vayan concretándose en realidades próximas. Las reuniones del Club se celebrarán los miércoles, de 18 a 20 horas en «Sistema», Balmes, 434 - Teléfono (93) 211 54 40 - Barcelona, a donde pueden dirigirse todos los interesados en las actividades del nuevo Club.

TRUCOS

cambio de bases en una sola línea

por **P. MASATS**



```
CONVERSION EN UNA LINEA

READY.

100 IFDTHENA=INT(D/16):H$=MID$("01234567
89ABCDEF",1+D-A*16,1)+H$:D=A:GOTO100
200 D=0:IFH$>""THENFORI=1TOLEN(H$):A=ASC
(MID$(H$,I,1))-48:D=D*16+A+(A>9)*7:NEXT

CONVERSION DECIMAL A BINARIO

READY.

10 INPUTD:H$=""
100 IFDTHENA=INT(D/2):H$=MID$("01",1+D-A
*2,1)+H$:D=A:GOTO100
101 PRINTH$
```

Existen serias dudas filosóficas acerca de si un programa puede llamarse así cuando está compuesto de una sola línea pero, de hecho, la ejecución de la línea 100 del siguiente listado nos da en la variable H$ el valor de notación hexadecimal del valor existente en la variable D. Del mismo modo, la línea 200 — ella solita — nos hace la conversión contraria. Sólo debe tenerse en cuenta que, antes de ejecutarse la línea 100, la variable H$ debe estar vacía (H$ = ""). Y esto no es todo: sustituyendo los valores 16 y conformando adecuadamente la cadena de signos de la línea 100, se puede realizar la conversión de decimal a cualquier base. En la parte superior se da un ejemplo de conversión decimal-binario donde además se ilustra el manejo del «programa».

MARKETCLUB

- **Vendo aplicación de facturación con control de representantes, 9 listados, 6 ficheros, estadísticas, etc., permite copias de seguridad. Configuración: VIC, 8K, disco e impresora, 40.000 Ptas. Escribir a Jaime Ameller Pons. General Mola, 15, 1º B. CALATAYUD (Zaragoza).**

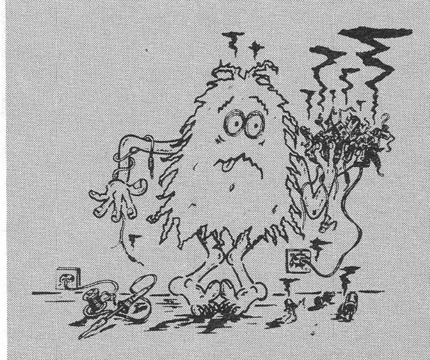
- **Vendo equipo CBM 3032 y 3040 (CPU y Floppy). Interesados llamar a Miguel al (93) 300 53 21 de BARCELONA.**

- **Vendo interfaz y programa para RTTY y CW para el PET a 15 K. Rafael, EA3CGK - Avda. Barcelona, 21, A, 4º 2ª. IGUALADA (Barcelona).**

- **Se busca experto en VIC-20 para colaborar en la creación y coordinación de un Club de Usuarios de VIC en Barcelona. Llamar a Srta. Rosa Romero. Teléfono 211 54 40.**

- **Vendo cartucho 16K VIC-20, por 14.000 ptas. Hago programas en Basic de Commodore (todas las versiones) bajo encargo. Desearía contactar con usuarios de Commodore en la zona de Madrid, para cambio de programas, impresiones, pokes especiales, etc... Razón: Francisco Gutiérrez. Santiago Rusiñol, 12. MADRID-3. Teléf. (91) 253 13 40. Horas comida y cena.**

SOCORRO



el esquema del VIC-20

P. MASATS

Aunque en el número 3 ya se explicó cuál será el contenido de esta sección recordaremos para los despistados y los recién llegados (¡por cierto: BIENVENIDOS!) a la comunidad de lectores de nuestra Revista, que aquí se tratará de responder a estos pequeños problemas de HARDWARE que tan enojosos son a menudo. Para empezar pocas cosas hay mejores que el propio esquema del VIC-20. Para mejorar al máximo su legibilidad lo daremos en tres partes a partir de este número. Por cierto que, dada la cantidad de cartas pidiendo este esquema, el título de esta sección hemos estado a punto de aplicárnoslo a nosotros. Esperamos que este interés se materialice en un buen número de realizaciones por parte de los usuarios que se traducirán, mediante la correspondiente colaboración, en artículos para la Revista. (Si alguien había supuesto que nos habíamos olvidado de la paliza de las colaboraciones, ya puede irse enterando de que no es así).

Aprovechamos la ocasión para dar algunas indicaciones sobre causas de problemas en el VIC-20: Si los cables de la unidad de cassette y el que va al televisor están demasiado juntos, pueden producirse errores en la grabación y lectura de programas causados por las tensiones inducidas en el cable de señal del grabador por las altas frecuencias implicadas en la señal de video. Por las mismas razones agravadas por las corrientes inducidas — de mayor magnitud —, en el interior de un televisor normal pueden darse errores de lectura y escritura si se trabaja con la unidad de cassette situada encima del televisor.

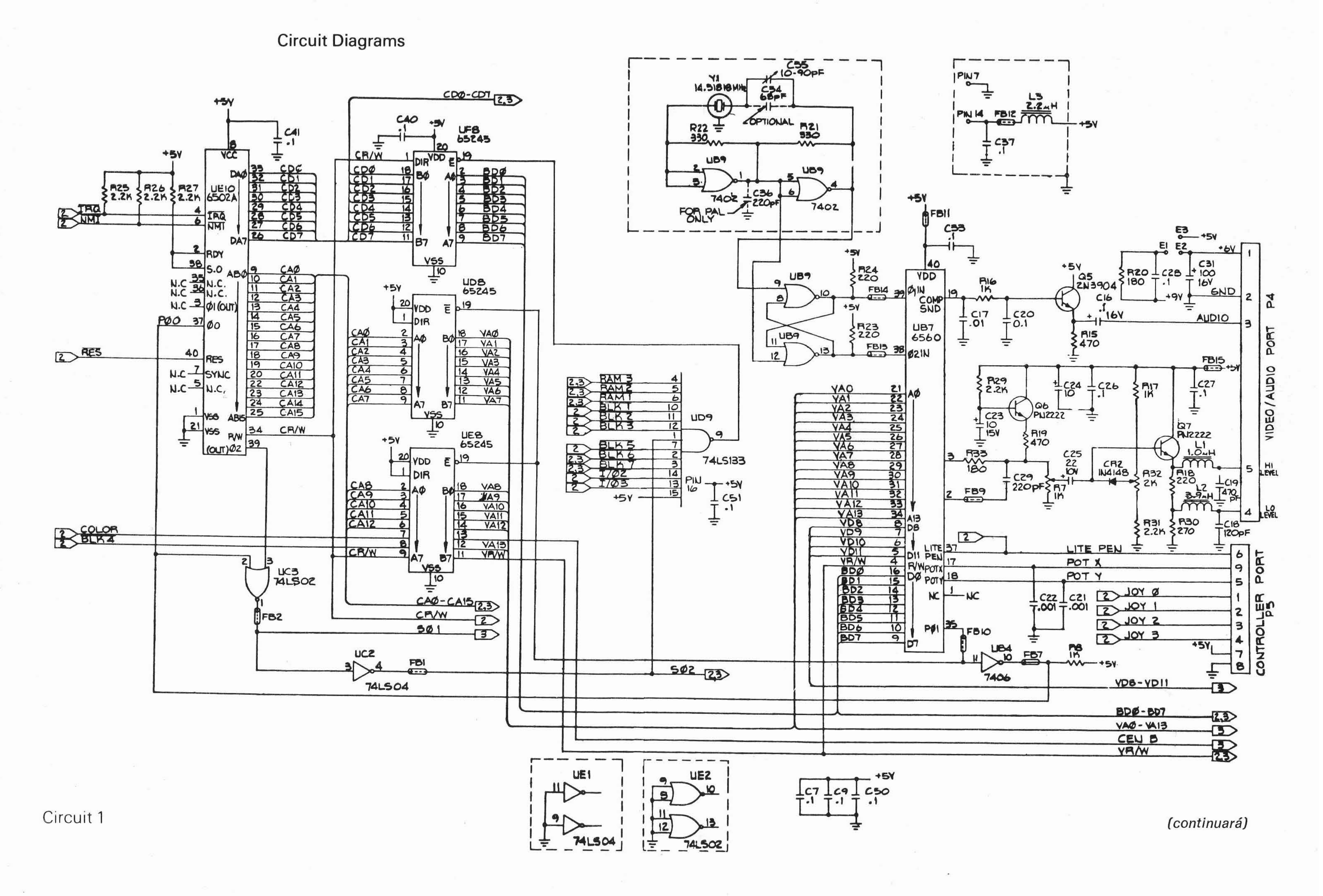
ÚLTIMA HORA

la segunda parte del Curso de Introducción al BASIC

está a punto de aparecer

En estos momentos se está terminando el trabajo de traducción e impresión de la segunda parte del Curso de Introducción al Lenguaje de Programación BASIC, por lo que, en breve tiempo, se podrá disponer de los primeros ejemplares. En el próximo número daremos más detalles.

Circuit Diagrams



Circuit 1

(continuará)

VIC-20

EL ORDENADOR PERSONAL AMPLIABLE CON COLOR Y SONIDO.



49.500 Ptas.
COLOR-SONIDO

Así es el VIC-20

- Lenguaje BASIC extendido.
- Sistema operativo COMMODORE.
- 5 K RAM ampliable a 32 K.
- 16 colores, 4 generadores de sonido.
- 66 caracteres gráficos.
- Periféricos disponibles:
  - Cassette.
  - Impresora de agujas.
  - Unidad de disco de 170 K.

Así hace las cosas el VIC-20

- Enseña informática.
- Efectúa todo tipo de cálculos matemáticos.
- Realiza funciones docentes.
- Se encarga de múltiples tareas profesionales.
- Proporciona divertidos momentos de ocio.
- Ayuda a planificar labores domésticas.
- Hace todas las aplicaciones que Vd. imagine.



GRATIS

Con la adquisición de su VIC-20 recibirá además:
- MANUAL DEL USUARIO.
- INTRODUCCION AL LENGUAJE DE PROGRAMACION BASIC.
- Y 17 PROGRAMAS DE PRACTICAS (en dos cassettes).



commodore COMPUTER

Distribuidor exclusivo para España:

Microelectrónica y Control, S.A.
Taquígrafo Serra, 7 5.°. Barcelona-29
Princesa, 47 3.° G. Madrid-8

De venta en tiendas especializadas.

SLOGAN